

# Computer Vision: Algorithms and Applications

Richard Szeliski

September 7, 2009

# Chapter 1

## Introduction

1.1	A brief history . . . . .	11
1.2	Book overview . . . . .	18
1.3	Additional reading . . . . .	27



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.*

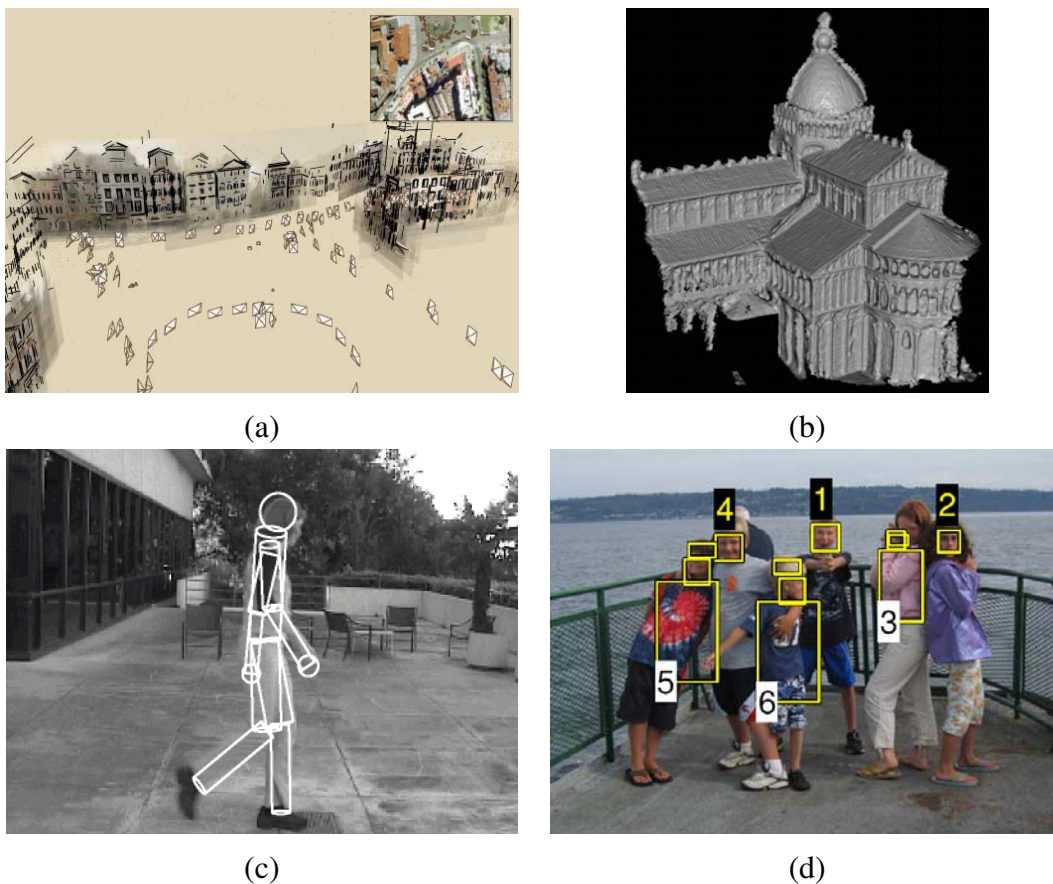


Figure 1.2: *Some examples of computer vision algorithms and applications. (a) Structure from motion algorithms can reconstruct a sparse 3D point model of a large complex scene from hundreds of partially overlapping photographs (Snavely et al. 2006). (b) Stereo matching algorithms can build a detailed 3D model of a building façade from hundreds of differently exposed photographs taken from the Internet (Goesele et al. 2007). (c) Person tracking algorithms can track a person walking in front of a cluttered background (Sidenbladh and Black 2003). (d) Face detection algorithms, coupled with color-based clothing and hair detection algorithms, can locate and recognize the individuals in this image (Sivic et al. 2006).*

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). Or, 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 appearance. Perceptual psychologists have spent decades trying to understand how the visual system works, and even though they can devise optical illusions to tease apart some of its principles (Figure 1.3), a complete solution to this puzzle remains elusive (Marr 1982, Palmer 1999, Livingstone 2008).

Researchers in computer vision have in parallel been developing mathematical techniques for recovering the three-dimensional shape and appearance of objects in imagery. We now have reliable techniques for accurately computing a partial 3D model of an environment from thousands of partially overlapping photographs (Snavely *et al.* 2006) (Figure 1.2a). Given a large enough set of views of a particular object or façade, we can create accurate dense 3D surface models using stereo matching (Goesele *et al.* 2007) (Figure 1.2b). We can track a person moving against a complex background (Sidenbladh and Black 2003) (Figure 1.2c). We can even, with moderate success, attempt to find and name all of the people in a photograph using a combination of face, clothing, and hair detection and recognition (Sivic *et al.* 2006) (Figure 1.2d). However, despite all of these advances, the dream of having a computer interpret an image at the same level as a two-year old (say counting all of the animals in a picture) remains elusive.

Why is vision so difficult? In part, it's because vision 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* 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, 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 (no fully computer-animated movie with human characters has yet succeeded at crossing the *uncanny valley* that separates real humans from android robots and computer animated humans<sup>1</sup>), in limited domains, such as rendering a still scene composed of everyday objects, or even animating extinct creatures such as dinosaurs, the illusion of reality is

---

<sup>1</sup> The term *uncanny valley* was originally coined by roboticist Masahiro Mori as applied to robotics (Mori 1970). It is also commonly applied to computer animated films such as *Final Fantasy* and *Polar Express* ([http://en.wikipedia.org/wiki/Uncanny\\_Valley](http://en.wikipedia.org/wiki/Uncanny_Valley)).

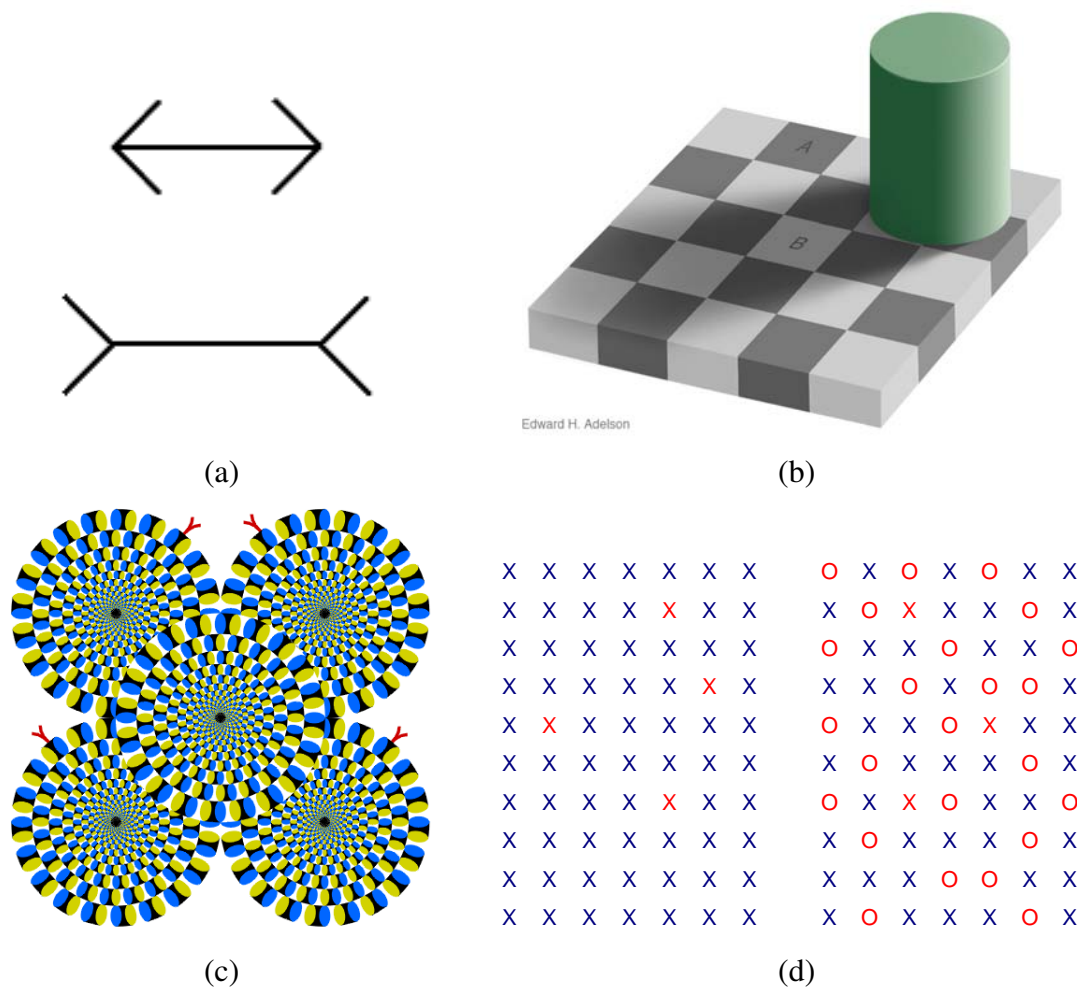


Figure 1.3: Some common optical illusions and what they might tell us about the visual system. (a) The classic Müller-Lyer illusion, where the length of the two horizontal lines appear different, probably due to the imagined perspective effects, [http://www.michaelbach.de/ot/sze\\_muelue](http://www.michaelbach.de/ot/sze_muelue). (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 (from [http://web.mit.edu/persci/people/adelson/checkershadow\\_illusion.html](http://web.mit.edu/persci/people/adelson/checkershadow_illusion.html)). (c) The static image appears to move in a consistent direction, possibly due to differences in transmission speeds between different visual pathways (from <http://www.psy.ritsumeai.ac.jp/~akitaoka/rotsnakee.html>). (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.

[ Note: Acknowledge all figures and replace if cannot get permissions. ]

perfect.

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 (Figure 1.10). 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. (Colleagues at work often ask me for software to find and name all the people in photos, so they can get on with the more “interesting” work. :-) This misperception that vision should be easy dates back to the early days of artificial intelligence §1.1, when it was initially believed that the *cognitive* (logic proving, 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)<sup>2</sup>,
- **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 (Figure 1.4c),
- **3D model building (photogrammetry):** fully automated 3D model building from aerial photographs used in systems such as Virtual Earth<sup>3</sup>,
- **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;
- **Automotive safety:** detecting unexpected obstacles such as pedestrians on the street, under conditions where active vision techniques such as radar or lidar do not work as well (Figure 1.4e);
- **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 the Hollywood (e.g., in movies such as Jurassic Park) (Roble 1999), and also require the use of precise *matting* to insert new elements between foreground and background elements (Chuang *et al.* 2002).

---

<sup>2</sup> [http://en.wikipedia.org/wiki/Automatic\\_number\\_plate\\_recognition](http://en.wikipedia.org/wiki/Automatic_number_plate_recognition)

<sup>3</sup> <http://bing.com/maps>

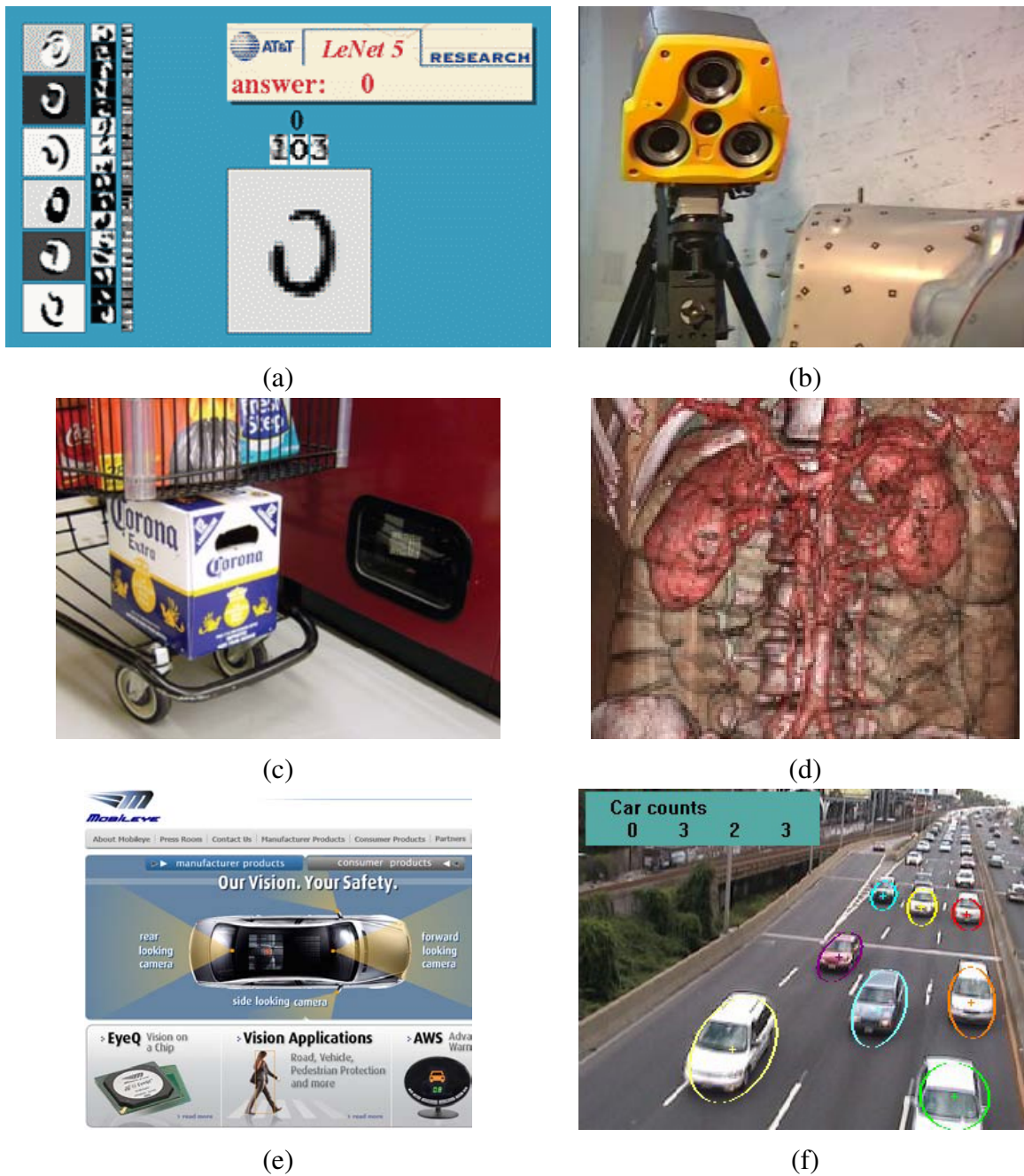


Figure 1.4: Some industrial applications of computer vision <http://www.cs.ubc.ca/spider/lowe/vision.html>: (a) optical character recognition (OCR) <http://yann.lecun.com/exdb/lenet/>; (b) machine inspection <http://www.cognitens.com/>; (c) retail <http://www.evoretail.com/>; (d) medical imaging <http://www.clarontech.com/>; (e) automotive safety <http://www.mobileye.com/>; (f) surveillance and traffic monitoring <http://www.honeywellvideo.com/>.

- **Motion capture (mocap):** of actors for computer animation, using retro-reflective markers viewed from multiple cameras or other vision-based techniques<sup>4</sup>,
- **Surveillance:** monitoring for intruders, analyzing highway traffic (Figure 1.4f), monitoring pools for drowning victims,
- **Fingerprint recognition:** for automatic access authentication as well as forensic applications,
- **Other applications:** David Lowe’s web site of industrial vision applications, <http://www.cs.ubc.ca/spider/lowe/vision.html>, lists many other interesting industrial application of computer vision.

While these are all extremely important applications, they mostly pertain to fairly specialized kinds of imagery and narrow domains.

In this book, we focus more on broader *consumer-level* applications, such as fun things you can do with your own personal photographs and video. These include:

- **Stitching:** turning overlapping photos into a single seamlessly stitched panorama, as described in §9 (Figure 1.5a);
- **Exposure bracketing:** merge multiple exposures taken under challenging lighting conditions (strong sunlight and shadows) into a single perfectly exposed image §10.2 (Figure 1.5b);
- **Morphing:** turning one of your friend’s pictures into another one’s, using a seamless *morph* transition §8.3.2 (Figure 1.5c);
- **3D modeling:** convert one or more snapshots into a 3D model of the object or person you are photographing §12.6 (Figure 1.5d);
- **Video match move and stabilization:** insert 2D pictures or 3D models into your videos by automatically tracking nearby reference points<sup>5</sup> §7.4.2, or use the motion estimates to remove shake from your videos §8.2.1;
- **Photo-based walkthroughs:** navigate a large collection of photographs, such as the interior of your house, by flying between different photos in 3D (§13.1.2 and §13.5.5)
- **Face detection:** for improved camera focusing as well as more relevant image search §14.2;

---

<sup>4</sup> [http://en.wikipedia.org/wiki/Motion\\_capture](http://en.wikipedia.org/wiki/Motion_capture)

<sup>5</sup> For a fun student project on this topic, see the “PhotoBook” project at <http://www.cc.gatech.edu/dvfx/videos/dvfx2005.html>.



- **Visual authentication:** automatically log different family members onto your home computer as they sit down in front of the Web cam §14.1.

The great thing about these applications is that they are already familiar to most students, or at least they are technologies that they 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<sup>6</sup> 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 these can be used to *formulate* and *constrain* the potentially open-ended problems endemic in vision. For example, if someone comes to me and asks for a good edge detector, my first question is usually to ask *why?* What kind of problem are they trying to solve, and why do they believe that edge detection is an important component?

If they are trying to locate faces §14.2, I usually point out that most successful face detectors use a combination of skin color detection (Exercise 2.9) and simple blob features §14.2, and do not rely on edge detection.

If they are trying to match door and window edges in a building for the purpose of 3D reconstruction, I tell them that edges are a fine idea, but it's better to tune the edge detector for long edges §3.2.1, §4.2, and then link these together into straight lines with common vanishing points before matching §4.3.

Thus, it's better to think back from 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/or specifications for the problem. Then, I try to find out which techniques are known to work, implement a few of these, and finally evaluate their performance and make a selection. In order for this process to work, it's 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 get used.

However, this book is not just an engineering text (a source of recipes). It also takes a *scientific* approach to the 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.

---

<sup>6</sup> These techniques include physics, Euclidean and projective geometry, statistics, and optimization, and make computer vision a fascinating field to study and a great way to learn techniques widely applicable in other fields.

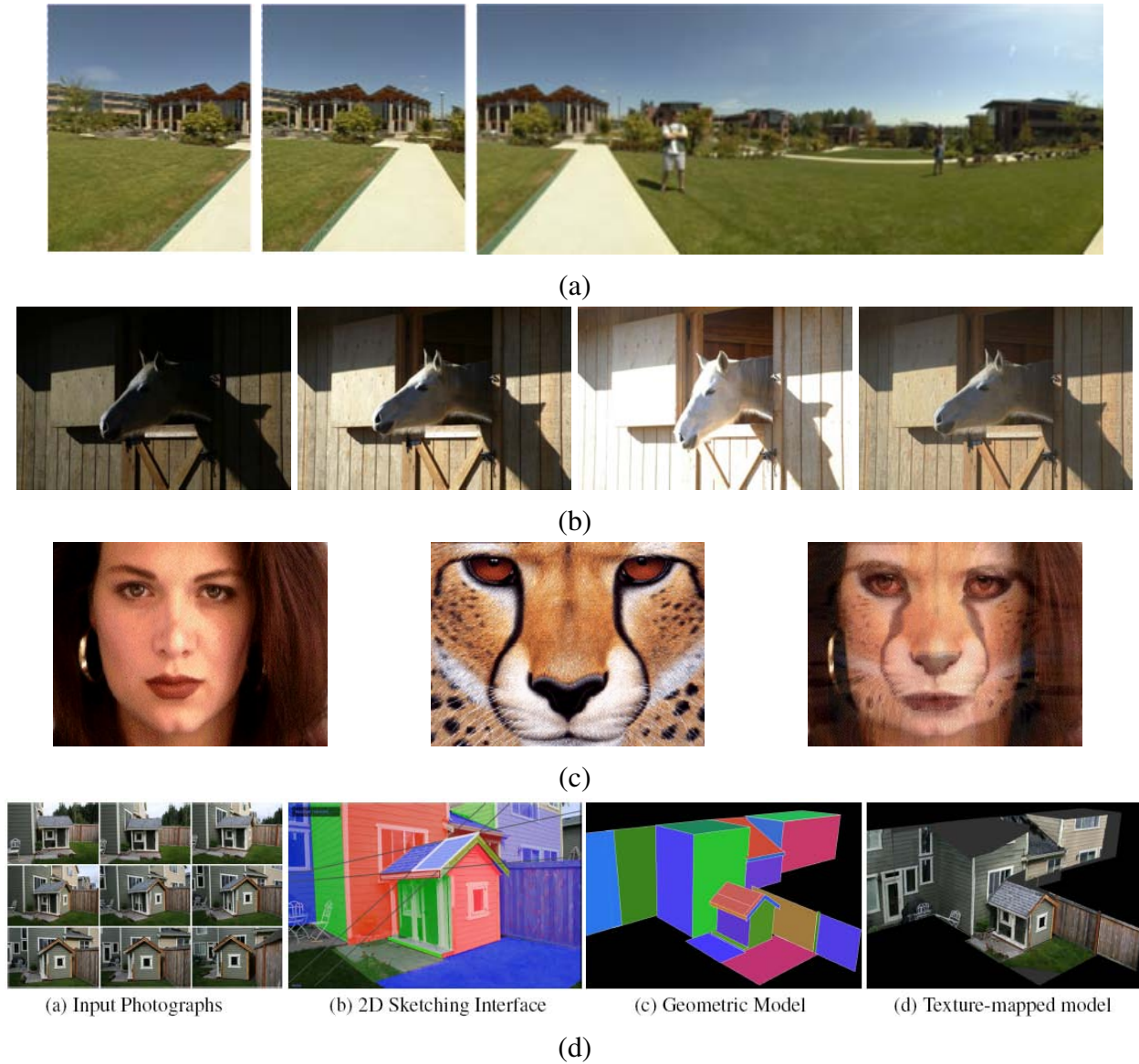


Figure 1.5: *Some consumer applications of computer vision: (a) image stitching: merging different views; (b) exposure bracketing: merging different exposures; (c) morphing: blending between two photographs; (d) turning a collection of photographs into a 3D model.*

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 even possible to associate a risk or loss function with mis-estimating the answer §B.2, and setting up your inference algorithm to minimize the expected risk. (Consider a robot trying to estimate the distance to an obstacle: it's 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 also 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 both algorithms that are *robust* to noise and deviation from our models, as well as reasonably *efficient* in terms of run-time 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.

Now that I've described the goals of this book and the frameworks that I use, let me devote the rest of this chapter to two additional topics. The first is a brief synopsis of the history of computer vision, §1.1. This 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, §1.2, which is useful reading for everyone who intends to make a study of this topic (or to jump in partway, since it describes inter-chapter dependencies). 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.

To support the book's use as a textbook, the appendices and associated web site contain additional course materials, including slide sets, test images and solutions, and pointers to software. The book chapters have exercises, some suitable as written homework assignments, others as shorter one-week projects, and still others as open-ended research problems suitable as challenging final projects.

As a reference book, I try wherever possible to discuss which techniques and algorithms work well in practice, as well as provide up-to-date pointers to the latest research results in the areas that I cover. The exercises can also be used to build up your own personal library of self-tested and validated vision algorithms, which in the long term (assuming you have the time), is more worthwhile than simply pulling algorithms out of a library whose performance you do not really

understand.

## 1.1 A brief history

[ Note: This whole section could definitely use some sanity checking from other researchers who know the older literature better. ]

In this section, I provide a brief personal synopsis of the main development in computer vision over the last thirty years, or at least those that I find personally interesting and which appear to 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 §1.2.

**1970s.** When computer vision first started out in the early 1970s, it was viewed as the visual perception component of an ambitious agenda to mimic human intelligence and to endow robots with intelligent behavior. At the time, it was believed by some of the early pioneers of artificial intelligence and robotics (at places like MIT, Stanford, and CMU) that solving the “visual input” problem would be an easy step along the path to solving more difficult problems such as higher-level reasoning and planning. According to one well-known story, in 1966, Marvin Minsky at MIT asked his undergraduate student Gerald Jay Sussman to “spend the summer linking a camera to a computer and getting the computer to describe what it saw” (Boden 2006, p. 781).<sup>7</sup> We now know that the problem is slightly more difficult than that.

What distinguished computer vision from the already existing field of digital image processing (Rosenfeld and Kak 1976) was a desire to recover the three-dimensional structure of the world from images, and to use this as a stepping stone towards full scene understanding. Winston (1975) and Hanson and Riseman (1978) provide two nice collections of classic papers from this early period.

Early attempt at scene understanding involved extracting edges and then inferring the 3D structure of an object from the topological structure of the 2D lines (Roberts 1965). Several *line labeling* algorithms were developed at that time (Huffman 1971, Clowes 1971, Waltz 1975, Rosenfeld *et al.* 1976, Kanade 1980) (Figure 1.6a). Nalwa (1993) gives a nice review of this area. The topic of edge detection §4.2 was also an active area of research; a nice survey on contemporaneous work can be found in (Davis 1975).

Three-dimensional modeling of non-polyhedral objects was also being studied (Baumgart 1974, Baker 1977). One popular approach used *generalized cylinders*, i.e., solid of revolutions and swept closed curves (Agin and Binford 1976, Nevatia and Binford 1977) often arranged into parts rela-

---

<sup>7</sup> Boden (2006) cites (Crevier 1993) as the original source.

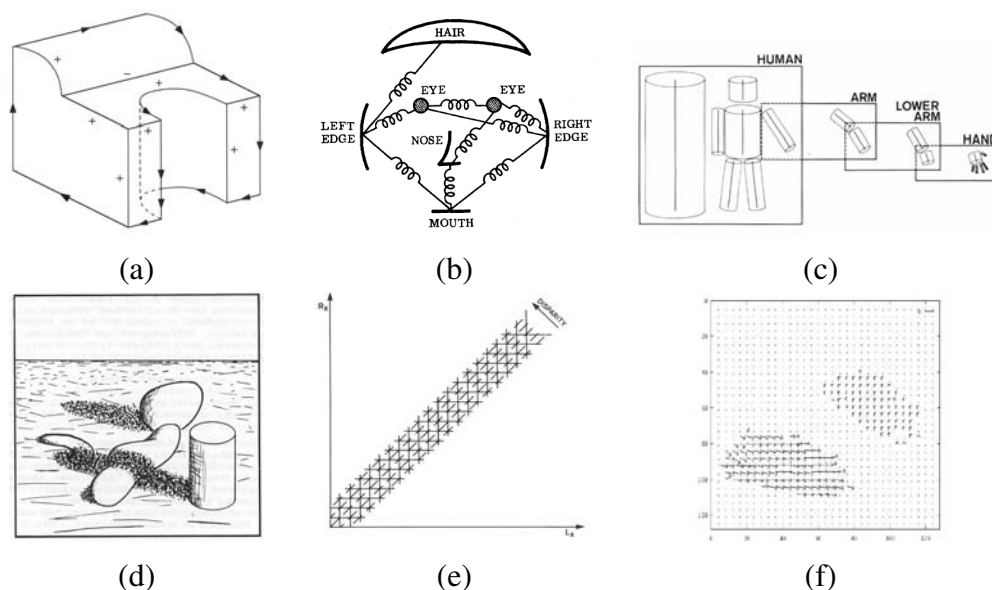


Figure 1.6: Some early examples of computer vision algorithms (1970s): (a) line labeling (Nalwa 1993), (b) pictorial structures (Fischler and Elschlager 1973), (c) articulated body model (Marr 1982), (d) intrinsic images (Barrow and Tenenbaum 1981), (e) stereo correspondence (Marr 1982), (f) optical flow (Nagel and Enkelmann 1986).

tionship<sup>8</sup> (Hinton 1977, Marr 1982) (Figure 1.6c). Fischler and Elschlager (1973) called such elastic arrangements of parts *pictorial structures* (Figure 1.6b). These are currently one of the favored approaches being used in object recognition §14.4 (Felzenszwalb and Huttenlocher 2005).

More quantitative approaches to computer vision were also developed at the time, including the first of many feature-based stereo correspondence algorithms §11.2 (Dev 1974, Marr and Poggio 1976, Moravec 1977, Marr and Poggio 1979, Mayhew and Frisby 1981, Baker 1982, Barnard and Fischler 1982, Ohta and Kanade 1985, Grimson 1985, Pollard *et al.* 1985, Prazdny 1985) (Figure 1.6e), as well as intensity-based optical flow algorithms §8.4 (Horn and Schunck 1981, Huang 1981, Lucas and Kanade 1981, Nagel 1986) (Figure 1.6f). The early work in simultaneously recovering 3D structure and camera motion §7 also began around this time (Ullman 1979, Longuet-Higgins 1981).

A more qualitative approach to understanding intensities and shading variations, and explaining these by the effects of image formation phenomena such as surface orientation and shadows was championed by (Barrow and Tenenbaum 1981) in their paper on *intrinsic images* (Figure 1.6d), along with Marr's (1982) related  $2\frac{1}{2}$ -*D sketch* ideas. This approach is again seeing a bit of a revival in the work of Tappen *et al.* (2005).

A lot of the philosophy of how vision was believed to work at the time is summarized in

<sup>8</sup> In robotics and computer animation, these linked part graphs are often called *kinematic chains*.

David Marr's (1982) book *Vision*. (More recent developments in visual perception theory are covered in (Palmer 1999, Livingstone 2008).) In particular, Marr introduced his notion of the three levels of description of a (visual) information processing system. These three levels, very loosely paraphrased according to my own interpretation, are:

- **Computational theory:** What is the goal of the computation (task) and what are the constraints that are known or can be brought to bear on the problem?
- **Representations and algorithms:** How are the input, output, and intermediate information represented, and which algorithms are used to calculate the desired result?
- **Hardware implementation:** How are the representations and algorithms mapped onto actual hardware, e.g., a biological vision system or a specialized piece of silicon? Conversely, how can hardware constraints be used to guide the choice of representation and algorithm? With the increasing use of graphics chips (GPUs) and many-core architectures for computer vision §C.2, this question is again becoming quite relevant.

As I mentioned earlier in this introduction, it is my conviction that a careful analysis of the problem specification and known constraints from image formation and priors (the scientific and statistical approaches) must be married with efficient and robust algorithms (the engineering approach) to design successful vision algorithms. Thus, it seems that Marr's philosophy is as good a guide to framing and solving problems in our field today as it was twenty-five years ago.

**1980s.** In the 1980s, a lot more attention was focused on more sophisticated mathematical techniques for performing quantitative image and scene analysis.

Image pyramids §3.4 started being widely used to perform tasks such as image blending and coarse-to-fine correspondence search (Rosenfeld 1980, Burt and Adelson 1983a, Burt and Adelson 1983b, Rosenfeld 1984, Quam 1984, Anandan 1989) (Figure 1.7a). Continuous version of pyramid using the concept of *scale-space* processing were also developed (Witkin 1983, Witkin *et al.* 1986, Lindeberg 1990). In the late 80s, wavelets §3.4.3 started displacing or augmenting regular image pyramids in some applications (Adelson *et al.* 1987, Mallat 1989, Simoncelli and Adelson 1990a, Simoncelli and Adelson 1990b, Simoncelli *et al.* 1992).

The use of stereo as a quantitative shape cue was extended by a wide variety of *shape-from-X* techniques, including shape from shading §12.1.1 (Horn 1975, Pentland 1984, Blake *et al.* 1985, Horn and Brooks 1986, Horn and Brooks 1989) (Figure 1.7b), photometric stereo §12.1.1 (Woodham 1981) shape from texture §12.1.2 (Witkin 1981, Pentland 1984, Malik and Rosenholtz 1997), and shape from focus §12.1.3 (Nayar *et al.* 1995). Horn (1986) has a nice discussion of most of these techniques.

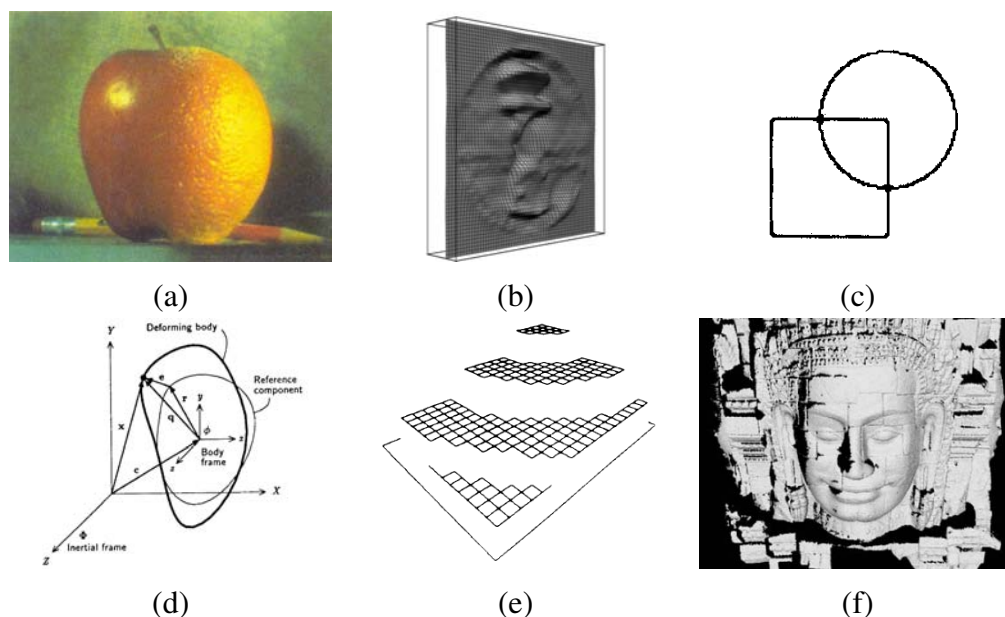


Figure 1.7: Examples of computer vision algorithms from the 1980s: (a) pyramid blending (Burt and Adelson 1983b), (b) shape from shading (Freeman and Adelson 1991), (c) edge detection (Freeman and Adelson 1991), (d) physically-based models (Terzopoulos and Witkin 1988), (e) regularization-based surface reconstruction (Terzopoulos 1988), (f) range data acquisition and merging (Banno et al. 2008).

Research into better edge and contour detection §4.2 was also active during this period, (Canny 1986, Nalwa and Binford 1986) (Figure 1.7c), including the introduction of dynamically evolving contour trackers §5.1.1 such as *snakes* (Kass et al. 1988), as well as three-dimensional *physically-based models* (Terzopoulos et al. 1987, Kass et al. 1988, Terzopoulos and Fleischer 1988, Terzopoulos et al. 1988) (Figure 1.7d).

Researchers noticed that a lot of the stereo, flow, shape-from-X, and edge detection algorithms could be unified, or at least described, using the same mathematical framework, if they were posed as variational optimization problems §3.6, and made more robust (well-posed) using regularization §3.6.1 (Terzopoulos 1983, Poggio et al. 1985, Terzopoulos 1986, Blake and Zisserman 1987, Bertero et al. 1988, Terzopoulos 1988). Around the same time, Geman and Geman (1984) pointed out that such problems could equally well be formulated using discrete *Markov Random Field* (MRF) models §3.6.2, which enabled the use of better (global) search and optimization algorithms such as simulated annealing.

Online variants of MRF algorithms that modeled and updated uncertainties using the Kalman filter were introduced a little later (Dickmanns and Graefe 1988, Matthies et al. 1989, Szeliski 1989). Attempts were also made to map both regularized and MRF algorithms onto parallel hard-

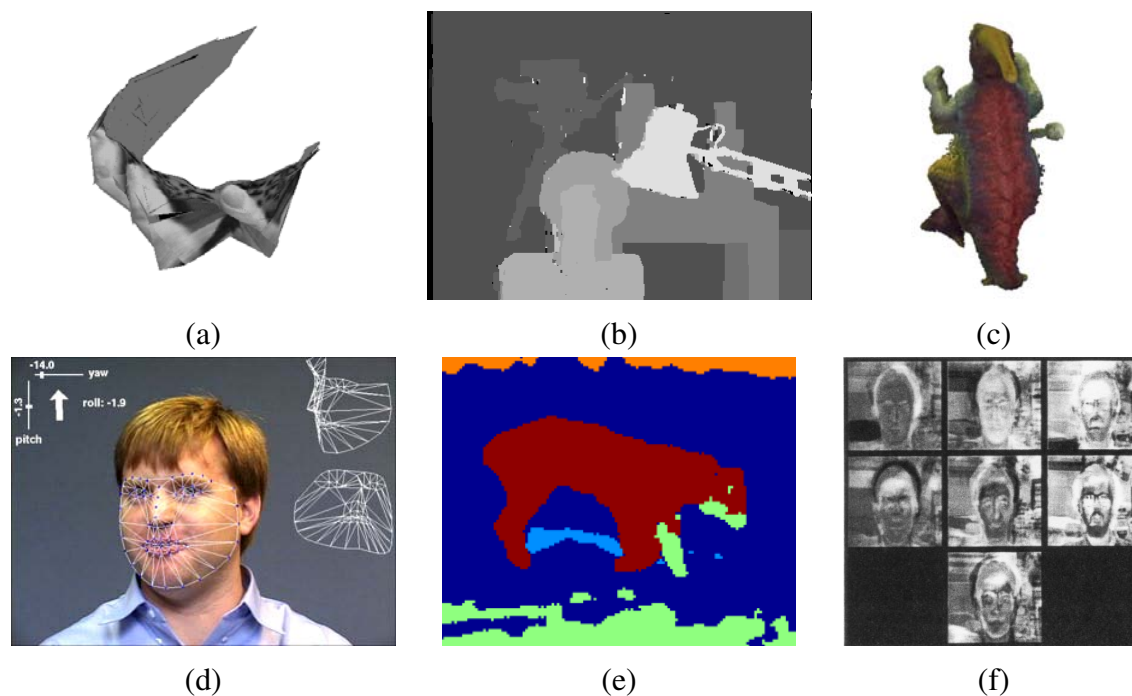


Figure 1.8: Examples of computer vision algorithms from the 1990s: (a) factorization-based structure from motion (Tomasi and Kanade 1992), (b) dense stereo matching (Boykov et al. 2001), (c) multi-view reconstruction (Seitz and Dyer 1999), (d) face tracking (Matthews and Baker 2004, Matthews et al. 2007), (e) image segmentation (Fowlkes et al. 2004), (f) face recognition (Turk and Pentland 1991a).

ware (Poggio and Koch 1985, Poggio et al. 1988a, Fischler et al. 1989). The book by Fischler and Firschein (1987) contains a very nice collection of articles focusing on all of these topics (stereo, flow, regularization, MRFs, and even higher-level vision).

Three-dimensional range data processing (acquisition, merging, modeling, and recognition) continued being actively explored during this decade (Agin and Binford 1976, Besl and Jain 1985, Faugeras and Hebert 1987, Curless and Levoy 1996) (Figure 1.7f). The compilation by Kanade (1987) contains a lot of the interesting papers in this area.

**1990s.** While a lot of the previously mentioned topics continued being explored, a few of them became significantly more active.

A burst of activity in using projective invariants for recognition (Mundy and Zisserman 1992) evolved into a concerted effort to solve the structure from motion problem §7. A lot of the initial activity was directed at *projective* reconstructions, which did not require the knowledge of camera calibration (Faugeras 1992, Hartley et al. 1992, Hartley 1994a, Faugeras and Luong 2001, Hartley and Zisserman 2004). Simultaneously, *factorization* techniques §7.3 were developed to ef-



ficiently solve problems for which orthographic camera approximations were applicable (Tomasi and Kanade 1992, Poelman and Kanade 1997, Anandan and Irani 2002) (Figure 1.8a) and then later extended to the perspective case (Christy and Horaud 1996, Triggs 1996). Eventually, the field started using full global optimization §7.4 (Taylor *et al.* 1991, Szeliski and Kang 1994, Azarbayejani and Pentland 1995), which was later recognized as being the same as the *bundle adjustment* techniques traditionally used in photogrammetry (Triggs *et al.* 1999). Fully automated (sparse) 3D modeling systems were built using such techniques (Beardsley *et al.* 1996, Schaffalitzky and Zisserman 2002, Brown and Lowe 2003, Snavely *et al.* 2006).

Work begun in the 1980s on using detailed measurements of color and intensity combined with accurate physical models of radiance transport and color image formation created its own subfield known as *physics-based vision*. A good survey of the field can be found in the three volume collection on this topic (Wolff *et al.* 1992a, Healey and Shafer 1992, Shafer *et al.* 1992).

Optical flow methods §8 continued being improved (Nagel and Enkelmann 1986, Bolles *et al.* 1987, Horn and Weldon Jr. 1988, Anandan 1989, Bergen *et al.* 1992a, Black and Anandan 1996, Bruhn *et al.* 2005, Papenberg *et al.* 2006), with (Nagel 1986, Barron *et al.* 1994, Baker *et al.* 2007) being good surveys. Similarly, a lot of progress was made on dense stereo correspondence algorithms §11 (Okutomi and Kanade 1993, Okutomi and Kanade 1994, Boykov *et al.* 1998, Birchfield and Tomasi 1999, Boykov *et al.* 2001) (see (Scharstein and Szeliski 2002) for a survey and comparison), with the biggest breakthrough being perhaps global optimization using *graph-cut* techniques (Boykov *et al.* 2001) (Figure 1.8b).

Multi-view stereo algorithms that produce complete 3D surfaces §11.6 were also an active topic of research (Seitz and Dyer 1999, Kutulakos and Seitz 2000) (Figure 1.8c) that continues being active today (Seitz *et al.* 2006). Techniques for producing 3D volumetric descriptions from binary silhouettes §11.6.2 continued being developed (Potmesil 1987, Srivasan *et al.* 1990, Szeliski 1993, Laurentini 1994), along with techniques based on tracking and reconstructing smooth occluding contours §11.2.1 (Cipolla and Blake 1992, Vaillant and Faugeras 1992, Zheng 1994, Boyer and Berger 1997, Szeliski and Weiss 1998, Cipolla and Giblin 2000).

Tracking algorithms also improved a lot, including contour tracking using *active contours* §5.1 such as *snakes* (Kass *et al.* 1988), *particle filters* (Blake and Isard 1998), and *level sets* (Malladi *et al.* 1995), as well as intensity-based (*direct*) techniques (Lucas and Kanade 1981, Shi and Tomasi 1994, Rehg and Kanade 1994), often applied to tracking faces (Lanitis *et al.* 1997, Matthews and Baker 2004, Matthews *et al.* 2007) (Figure 1.8d) and whole bodies (Sidenbladh and Black 2003, Hilton *et al.* 2006, Moeslund *et al.* 2006).

Image segmentation §5 (Figure 1.8e), a topic which has been active since the earliest days of computer vision (Brice and Fennema 1970, Horowitz and Pavlidis 1976, Riseman and Arbib 1977, Rosenfeld and Davis 1979, Haralick and Shapiro 1985, Pavlidis and Liow 1990), was also an active topic of research, including techniques based on minimum energy (Mumford and Shah 1989) and

minimum description length (Leclerc 1989), *normalized cuts* (Shi and Malik 2000), and *mean shift* (Comaniciu and Meer 2002).

Statistical learning techniques started appearing, first in the application of principal component *eigenface* analysis applied to face recognition §14.1.1 (Turk and Pentland 1991a) (Figure 1.8f) and linear dynamical systems for curve tracking §5.1.1 (Blake and Isard 1998).

Perhaps the most notable development in computer vision during this decade was the increased interaction with computer graphics (Seitz and Szeliski 1999), especially in the cross-disciplinary area of *image-based modeling and rendering* §13. The idea of manipulating real-world imagery directly to create new animations first came to prominence with *image morphing* techniques §3.5.3 (Beier and Neely 1992) (Figure 1.5c) and was later applied to *view interpolation* (Chen and Williams 1993, Seitz and Dyer 1996), panoramic image stitching §9 (Mann and Picard 1994, Chen 1995, Szeliski 1996, Szeliski and Shum 1997, Szeliski 2006a) (Figure 1.5a), and full light-field rendering §13.3 (Gortler *et al.* 1996, Levoy and Hanrahan 1996, Shade *et al.* 1998) (Figure 1.9a). At the same time, image-based modeling techniques for automatically creating realistic 3D models from collections of images were also being introduced (Beardsley *et al.* 1996, Debevec *et al.* 1996, Taylor *et al.* 1996) (Figure 1.9b).

**2000s.** This past decade has continued to see a deepening interplay between the vision and graphics fields. In particular, many of the topics introduced under the rubric of image-based rendering, such as image stitching §9, light-field capture and rendering §13.3, and *high dynamic range* (HDR) image capture through exposure bracketing §10.2 (Mann and Picard 1995, Debevec and Malik 1997) (Figure 1.5b), were re-christened as *computational photography* §10 to acknowledge the increased use of such techniques in everyday digital photography. For example, the rapid adoption of exposure bracketing to create high dynamic range images necessitated the development of *tone mapping* algorithms §10.2.1 to convert such images back to displayable results (Fattal *et al.* 2002, Durand and Dorsey 2002, Reinhard *et al.* 2002, Lischinski *et al.* 2006a) (Figure 1.9c). In addition to merging multiple exposures, techniques were developed to merge flash images with non-flash counterparts (Eisemann and Durand 2004, Petschnigg *et al.* 2004), and to interactively or automatically select different regions from overlapping images (Agarwala *et al.* 2004).

Texture synthesis §10.5 and quilting (Efros and Leung 1999, Efros and Freeman 2001, Kwatra *et al.* 2003) (Figure 1.9d) as well as in-painting (Bertalmio *et al.* 2000, Bertalmio *et al.* 2003, Criminisi *et al.* 2004) are two additional topics that can be classified as computational photography techniques, since they re-combine input image samples to produce novel photographs.

A second notable trend during this past decade was the emergence of feature-based techniques (combined with learning) for object recognition §14.3 (Ponce *et al.* 2007b). Some of the notable papers in this area include the *constellation model* of Fergus *et al.* (2003), Fergus *et al.* (2005), Fei-Fei *et al.* (2006) (Figure 1.9e) and the *pictorial structures* of Felzenszwalb and Huttenlocher

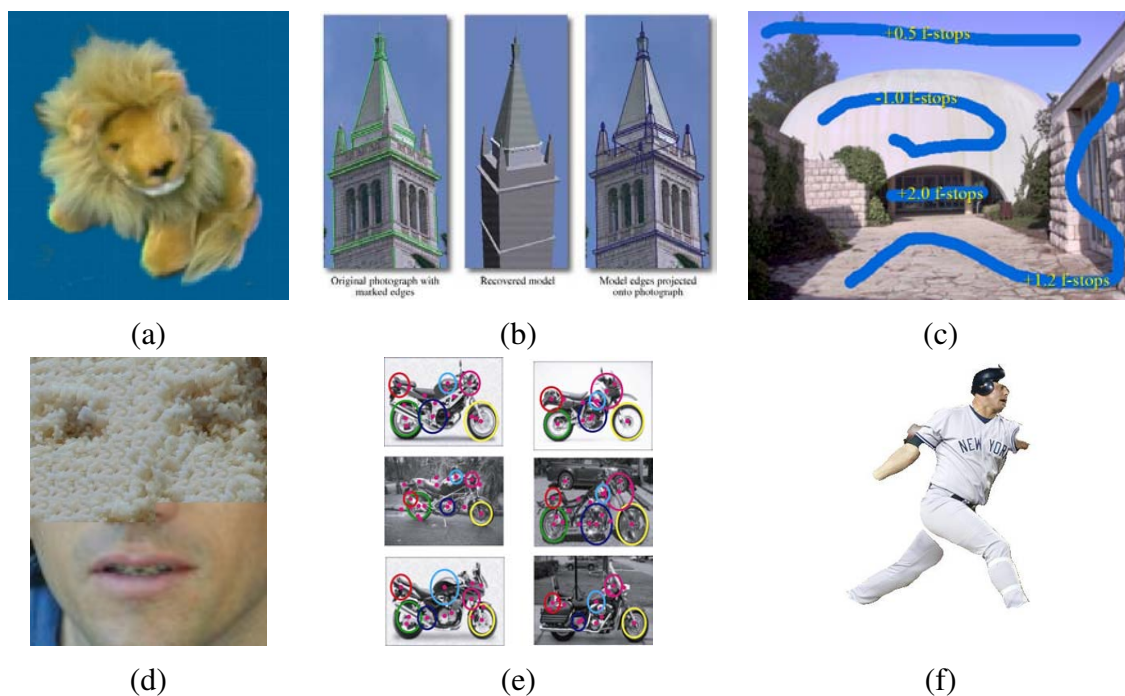


Figure 1.9: Recent examples of computer vision algorithms: (a) image-based rendering (Gortler et al. 1996), (b) image-based modeling (Debevec et al. 1996), (c) interactive tone mapping (Lischinski et al. 2006a), (d) texture synthesis (Efros and Freeman 2001), (e) feature-based recognition (Fergus et al. 2003), (f) region-based recognition (Mori et al. 2004).

(2005). Feature-based techniques also dominate other recognition tasks such as scene recognition (Zhang et al. 2007) and panorama and location recognition (Brown and Lowe 2007, Schindler et al. 2007). And while *interest point* (patch-based) features tend to dominate current research, some groups are pursuing recognition based on contours (Belongie et al. 2002) and region segmentation (Mori et al. 2004) (Figure 1.9f).

Another significant trend from this past decade has been the development of more efficient algorithms for complex global optimization problems §3.6, §B.6, (Szeliski et al. 2008c). While this trend began with work on graph cuts (Boykov et al. 2001, Kohli and Torr 2005), a lot of progress has also been made in message passing algorithms such as *loopy belief propagation* (LBP) (Yedidia et al. 2000, Kumar and Torr 2006).

## 1.2 Book overview

In this final part of this introduction, I give a brief tour of the material in this book, as well as a few notes on notation and some additional general references. Since computer vision is such a broad

field, it is possible to study certain aspects of it, e.g., geometric image formation and 3D structure recovery, without engaging other parts, e.g., the modeling of reflectance and shading. Some of the chapter in this book are only loosely coupled with others, and it is not strictly necessary to read all of the material in sequence.

Figure 1.10 shows a rough layout of the contents of this book. Since computer vision involves going from images to a structural description of the scene (and computer graphics the converse), I have positioned the chapters horizontally in terms of which major component they address, in addition to vertically, according to their dependence.

Going from left to right, we see the major column headings as Images (which are 2D in nature), Geometry (which encompasses 3D descriptions), and Photometry (which encompasses object appearance). (An alternative labeling for these latter two could also be *shape* and *appearance*—see, e.g., §13 and (Kang *et al.* 2000).) Going from top to bottom, we see increasing levels of modeling and abstraction, as well as techniques that build on previously developed algorithms. Of course, this taxonomy should be taken with a large grain of salt, as the processing and dependencies in this diagram are not strictly sequential, and subtle additional dependencies and relationships also exist (e.g., some recognition techniques make use of 3D information). The placement of topics along the horizontal axis should also be taken lightly, as most vision algorithms involve mapping between at least two different representations.<sup>9</sup> [ *Note: One could argue that photometry is actually closer to images than geometry, but since we usually go to geometry first, this seems easier to parse.* ]

Interspersed throughout the book are sample *Applications*, which relate the algorithms and mathematical material being presented in various chapters to useful, real-world applications. Many of these applications are also presented in the exercises sections, so that students can write their own.

At the end of each section, I provide a set of *Exercises* that the students can use to implement, test, and refine the algorithms and techniques presented in each section. Motivated students who implement a reasonable subset of these exercises will by the end of the book have a computer vision software library that can be used for a variety of interesting tasks and projects.

The book begins in Chapter 2 with a review of the image formation processes that create the images that we see and capture. Understanding this process is fundamental if you want to take a scientific (model-based) approach to computer vision. Students who are eager to just start implementing algorithms (or courses that have limited time) can skip ahead to the next chapter and dip into this material later.

In Chapter 2, we break down image formation into three major components. Geometric image formation §2.1 deals with points, lines, and planes, and how these are mapped onto images using *projective geometry* and other models (including radial lens distortion). Photometric image for-

---

<sup>9</sup> For an interesting comparison with what is known about the human visual system, e.g., the largely parallel *what* and *where* pathways, see some textbooks on human perception, e.g., (Palmer 1999, Livingstone 2008).

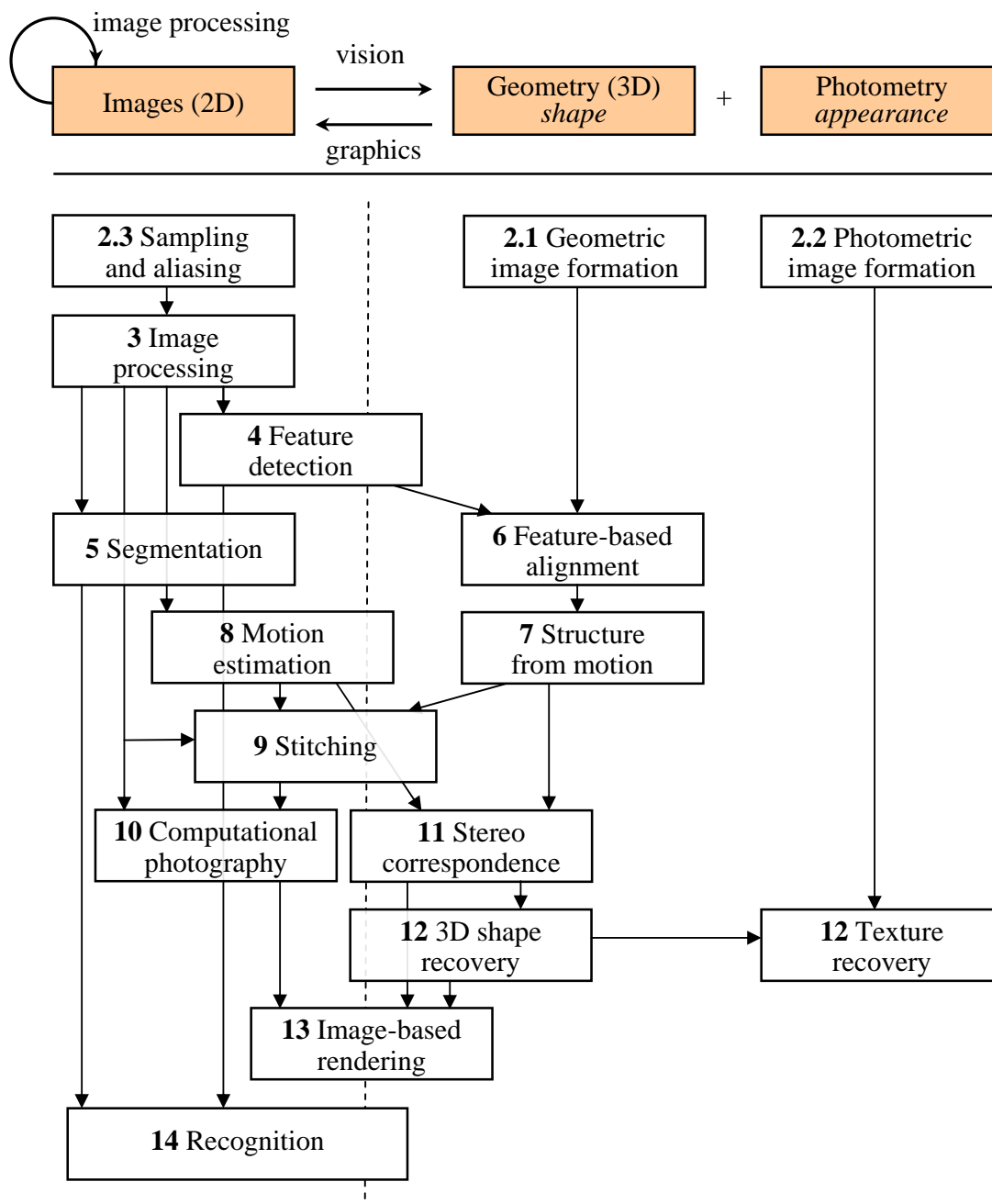


Figure 1.10: *Relationship between images, geometry, and photometry, as well as a taxonomy of the topics covered in this book. Topics are roughly positioned along the left-right axis depending on whether they are more closely related to image-based (left), geometry-based (middle) or appearance-based (right) representations, and on the vertical axis by increasing level of abstraction. The whole figure should be taken with a large grain of salt, as there are many additional subtle connections between topics not illustrated here.*

[ Note: Update this figure and caption if the structure of the book changes ]

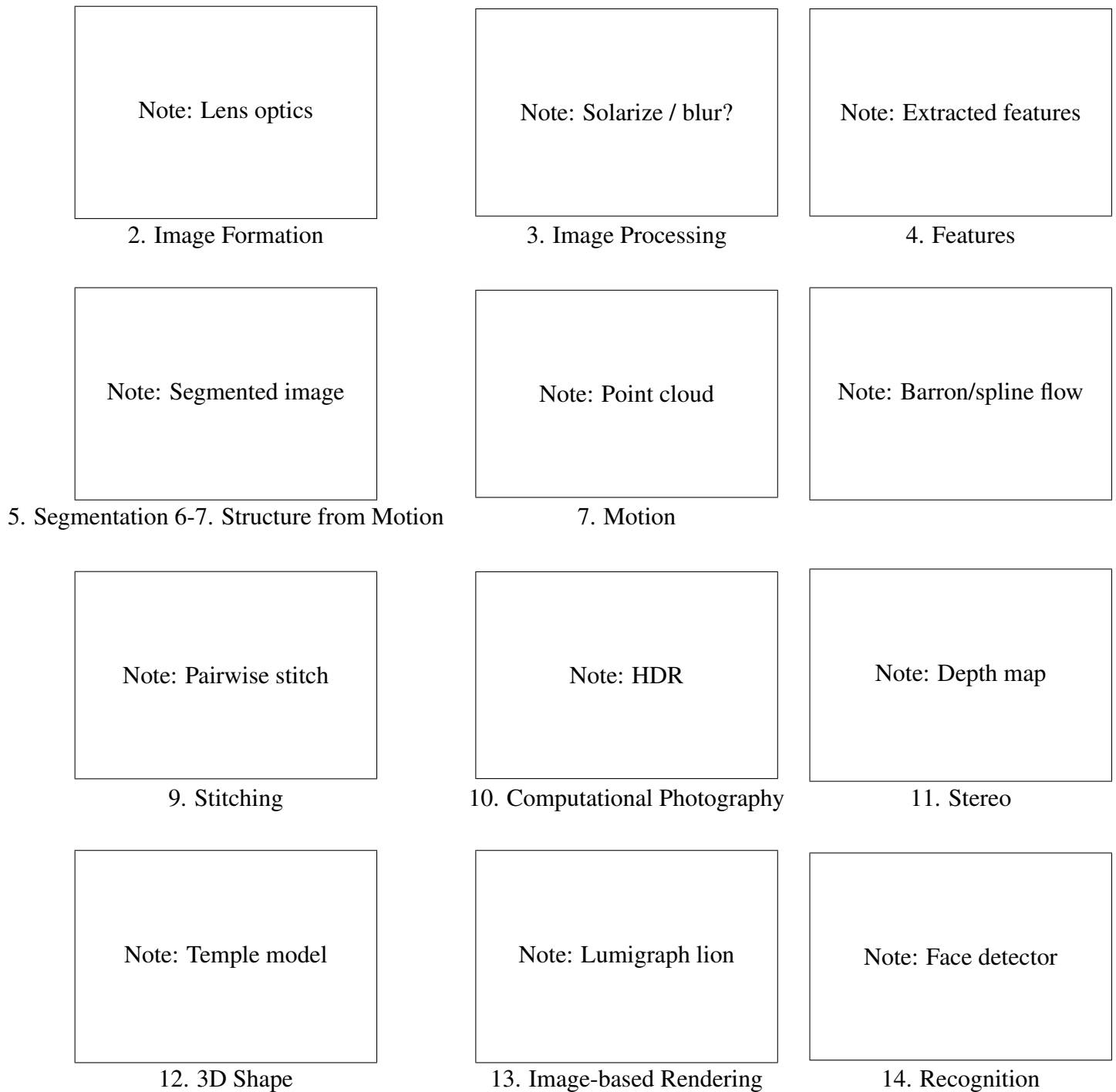


Figure 1.11: A pictorial summary of the chapter contents.

[ Note: Move this to earlier in the book, say after overview page? Or, incorporate into (an expanded) overview page, by associating a picture with each chapter, and listing section headings under each chapter name [Nice's Pen-Ink-Watercolor book]? Make sure it's consistent with book organization. ]

mation §2.2 covers *radiometry*, which describes how light interacts with surfaces in the world, as well as *optics*, which projects light onto the sensor plane. Finally, §2.3 covers how sensors work, including topics such as sampling and aliasing, color sensing, and in-camera compression.

Chapter 3 covers image processing, which is needed in almost all computer vision applications. This includes topics such as linear and non-linear filtering §3.2, the Fourier transform §3.3, image pyramids and wavelets §3.4, geometric transformations such as image warping §3.5, and global optimization techniques such as *regularization* and *Markov Random Fields* (MRFs) §3.6. While most of this material is covered in courses and textbooks on image processing, the use of optimization techniques is more typically associated with computer vision (although MRFs are now being widely used in image processing as well). The section on MRFs is also the first introduction to the use of Bayesian inference techniques, which are covered at a more abstract level in Appendix B. Chapter 3 also presents applications such as seamless image blending and image restoration.

Chapter 4 covers feature detection and matching. A lot of current 3D reconstruction and recognition techniques are built on extracting and matching *feature points* §4.1, so this is a fundamental technique required by many subsequent chapters (§6, §7, §9 and §14). We also cover edge and straight line detection §4.2–4.3.

Chapter 5 covers regions segmentation techniques, including active contour detection and tracking §5.1. Segmentation techniques include top-down (split) and bottom-up (merge) techniques, mean shift techniques, which find modes of clusters, and various graph-based segmentation approaches. All of these techniques are essential building blocks that are widely used in a variety of applications, including performance-driven animation, interactive image editing, and recognition.

Chapter 6 covers geometric alignment and camera calibration. We introduce the basic techniques of feature-based alignment in §6.1, and show how this problem can be solved using either linear or non-linear least squares, depending on the motion involved. We also introduce additional concept such as uncertainty weighting and robust regression, which are essential to making real-world systems work. Feature-based alignment is then used as a building block for 3D pose estimation (*extrinsic calibration*) §6.2 and camera (*intrinsic*) calibration §6.3. Chapter 6 also describes applications of these techniques to photo alignment for flip-book animations, 3D pose estimation from a hand-held camera, and single-view reconstruction of building models.

Chapter 7 covers the topic of *structure from motion*, which involves the simultaneous recovery of 3D camera motion and 3D scene structure from a collection of tracked 2D features. This chapter begins with the easier problem of 3D point *triangulation* §7.1, which is the 3D reconstruction of points from matched features when the camera positions are known. It then describes two-frame structure from motion §7.2, for which algebraic techniques exist, as well as robust sampling techniques such as RANSAC that can discount erroneous feature matches. The second half of Chapter 7 describes techniques for multi-frame structure from motion, including factorization §7.3, bundle

adjustment §7.4, and constrained motion and structure models §7.5. It also presents applications in view morphing, sparse 3D model construction, and match move.

In Chapter 8, we go back to a topic that directly deals with image intensities (as opposed to feature tracks), namely dense intensity-based motion estimation (*optical flow*). We start with the simplest possible motion models, namely translational motion §8.1, and cover topics such as hierarchical (coarse-to-fine) motion estimation, Fourier-based techniques, and iterative refinement. We then present parametric motion models, which can be used to compensate for camera rotation and zooming, as well as affine or planar perspective motion §8.2. This is then generalized to spline-based motion models §8.3 and then finally to general per-pixel optical flow §8.4, including layered and learned motion models §8.5. Applications of these techniques include automated morphing, frame interpolation (slow motion), and motion-based user interfaces.

Chapter 9 is devoted to *image stitching*, i.e., the construction of large panoramas and composites. While stitching is just one example of *computation photography* §10, there is enough depth here to warrant a separate chapter. We start by discussing various possible motion models §9.1, including planar motion and pure camera rotation. We then discuss global alignment §9.2, which is a special (simplified) case of general bundle adjustment, and then present *panorama recognition*, i.e., techniques for automatically discovering which images actually form overlapping panoramas. Finally, we cover the topics of *image compositing* and *blending* §9.3, which involve both selecting which pixels get used from which images, and blending them together so as to disguise exposure differences.

Image stitching is a wonderful application that ties together most of the material covered in earlier parts of this book. It also makes a good mid-term course project that can build on previously developed techniques such as image warping and feature detection and matching. Chapter 9 also presents more specialized variants of stitching such as whiteboard and document scanning, video summarization, *panography*, full 360° spherical panoramas, and interactive photomontage for blending repeated action shots together.

Chapter 10 presents additional examples of *computational photography*, which is the process of creating novel images from one or more input photographs, often based on the careful modeling and calibration of the image formation process §10.1. Computational photography techniques include merging multiple exposures to create *high dynamic range* images §10.2, increasing image resolution through blur removal and *super-resolution* §10.3, and image editing and compositing operations §10.4. We also cover the topics of texture analysis, synthesis and *inpainting* (hole filling) §10.5, as well as non-photorealistic rendering §10.5.2.

In chapter 11, we turn to the issue of stereo correspondence, which can be thought of as a special case of motion estimation where the camera positions are already known §11.1. This additional knowledge enables stereo algorithms to search over a much smaller space of correspondences, and in many cases to produce dense depth estimates that can be converted into visible surface mod-



els §11.3. We also cover multi-view stereo algorithms that build a true 3D surface representation instead of just a single depth map §11.6. Applications of stereo matching include head and gaze tracking, as well as depth-based background replacement (*Z-keying*).

Chapter 12 covers additional 3D shape and appearance modeling techniques. These include classic *shape-from-X* techniques such as shape from shading, shape from texture, and shape from focus §12.1, as well as shape from smooth occluding contours §11.2.1 and silhouettes §12.5. An alternative to all of these *passive* computer vision techniques is to use *active rangefinding* §12.2, i.e., to project patterned light onto the scenes and to recover the 3D geometry through triangulation. Processing all of these 3D representations often involves interpolating and/or simplifying the geometry §12.3, or using alternative representations such as surface point sets §12.4.

The collection of techniques for going from one or more images to partial or full 3D models is often called *image-based modeling* or *3D photography*. The next part of Chapter 12 examines three more specialized application areas (architecture, faces, and human bodies), which can use *model-based reconstruction* to fit parameterized models to the sensed data §12.6. The final part of Chapter 12 examines the topic of *appearance modeling* §12.7, i.e., techniques for estimating the texture maps, albedos, or even sometimes complete *bi-directional reflectance distribution functions* (BRDFs) that describe the appearance of 3D surfaces.

In Chapter 13, we discuss the large number of image-based rendering techniques that have been developed in the last decade, including simpler techniques such as view interpolation §13.1, layered depth images §13.2, and sprites and layers §13.2.2, as well as the more general framework of lightfields and Lumigraphs §13.3 and higher-order fields such as environment mattes §13.4.1. Applications of these techniques include navigating 3D collections of photographs using *Photo Tourism* and viewing 3D models as *object movies*.

In Chapter 13, we also discuss video-based rendering, which is the temporal extension of image-based rendering. The topics we cover include video-based animation §13.5.1, periodic video turned into *video textures* §13.5.2, and 3D video constructed from multiple video streams §13.5.4. Applications of these techniques include video denoising, morphing, and walkthroughs/tours based on 360° video.

Chapter 14 describes different approaches to recognition. It begins with techniques for detecting and recognizing faces §14.1–§14.2, then looks at techniques for finding and recognizing particular objects (*instance recognition*) §14.3. Next, we cover the most difficult variant of recognition, namely the recognition of broad *categories*, such as cars, motorcycles, horses, and other animals §14.4 and the role that scene context plays in recognition §14.5.

The book also contains three appendices with more detailed mathematical topics and additional material. Appendix A covers linear algebra and numerical techniques, including matrix algebra, least-squares, and iterative techniques. Appendix B covers Bayesian estimation theory, including maximum likelihood, robust statistics, Markov Random Fields, and uncertainty modeling.

Week	Material	Project
(1.)	§2 Image formation	
2.	§3 Image processing	
3.	§4 Feature detection and matching	P1
4.	§6 Projective geometry and alignment	
5.	§9 Image stitching and blending	P2
6.	§8 Optical flow and tracking	
7.	§7 Structure from motion	PP
8.	§14 Recognition	
(9.)	§10 More computational photography	
10.	§11 Stereo matching	
(11.)	§12 Multi-view stereo and 3D modeling	
12.	§13 Image-based rendering	
13.	Final project presentations	FP

Table 1.1: *Sample syllabi for 10-week and 13-week courses. The weeks in parentheses are not used in the shorter 10-week version. P1 and P2 are two early-term mini-projects, PP is when the (student selected) final project proposals are due, and FP is the final project presentations.*

Appendix C describes the supplementary material available to complement this book, including images and data sets, pointers to software, course slides, and an on-line bibliography.

## Sample syllabus

Teaching all of the material covered in this book in a single quarter or semester course is a Herculean task, and likely one not worth attempting. It is better to simply pick and choose topics related to the lecturer's preferred emphasis, as well as tailored to the set of mini-projects envisioned for the students.

Steve Seitz and I have successfully used a 10-week syllabus similar to the one shown in Table 1.1 (omitting the parenthesized weeks) as both an undergraduate and a graduate-level course in computer vision. The undergraduate course<sup>10</sup> tends to go lighter on the mathematics and takes more time reviewing basics, while the graduate level course<sup>11</sup> dives more deeply into techniques and assumes the students already have a decent grounding in either vision or related mathematical techniques. (See also the *Introduction to Computer Vision* course at Stanford<sup>12</sup>, which uses a

<sup>10</sup> <http://www.cs.washington.edu/education/courses/455/>

<sup>11</sup> <http://www.cs.washington.edu/education/courses/576/>

<sup>12</sup> <http://cs223b.stanford.edu/>

similar curriculum.) Related courses have also been taught on the topics of 3D Photography<sup>13</sup> and Computational Photography<sup>14</sup>.

When Steve and I teach the course, we prefer to give the students several small programming projects early in the course rather than focusing on written homework and/or quizzes. With a suitable choice of topics, it is possible for these projects to build on each other. For example, introducing feature matching early on can be used in a second assignment to do image alignment and stitching. Alternatively, direct (optical flow) techniques can be used to do the alignment, and more focus can be put on either graph cut seam selection or multi-resolution blending techniques.

We also ask the students to propose a final project (with a set of suggested topics for those who need ideas) by the middle of the course, and reserve the last week of the class for student presentations. With any luck, some of these final projects can actually turn into conference submissions!

No matter how you decide to structure the course, or how you choose to use this book, I encourage you to try at least a few small programming tasks to get a good feel for how vision techniques work, and when they do not. Better yet, pick topics that are fun and can be used on your own photographs, and try to push your creative boundaries to come up with surprising results.

## A note on notation

For better or worse, the notation found in computer vision and multi-view geometry textbooks tends to vary all over the map (Faugeras 1993, Hartley and Zisserman 2004, Girod *et al.* 2000, Faugeras and Luong 2001, Forsyth and Ponce 2003). In this book, I use the convention I first learned in my high school physics class (and later multi-variate calculus and computer graphics courses), which is that vectors  $\mathbf{v}$  are lower case bold, matrices  $\mathbf{M}$  are upper case bold, and scalars  $(T, s)$  are mixed case italic. Unless otherwise noted, vectors operate as column vector, i.e., they post-multiply matrices,  $\mathbf{M}\mathbf{v}$ , although they are sometime written as comma-separated parenthesized lists  $\mathbf{x} = (x, y)$  instead of bracketed column vectors  $\mathbf{x} = [x \ y]^T$ . Some commonly used matrices are  $\mathbf{R}$  for rotations,  $\mathbf{K}$  for calibration matrices, and  $\mathbf{I}$  for the identity. The List of Symbols page right after the Preface contains a list of the most commonly used symbols in this book.

*[ Note: Could also move this whole paragraph onto that page. ]*

*[ Note: Probably not worth mentioning the following:*

*For homogeneous coordinates §2.1, use  $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\mathbf{x}$  in  $\mathcal{P}^2$ .*

*Cross product operator in matrix form is denoted by  $[\ ]_{\times}$ . ]*

<sup>13</sup> <http://www.cs.washington.edu/education/courses/558/06sp/>

<sup>14</sup> <http://graphics.cs.cmu.edu/courses/15-463/>

## 1.3 Additional reading

This book attempts to be self-contained, so that students can implement the basic assignments and algorithms described here without the need for outside references. However, it does pre-suppose a general familiarity with basic concepts in linear algebra and numerical techniques, which are reviewed in Appendix A, and image processing, which is reviewed in Chapter 3.

Students who want to delve in more deeply into these topics can look in (Golub and Van Loan 1996) for matrix algebra and (Strang 1988) for linear algebra. In image processing, there are a number of popular references, including (Crane 1997, Gomes and Velho 1997, Jähne 1997, Pratt 2001, Gonzales and Woods 2002, Russ 2007). For computer graphics, popular texts include (Foley *et al.* 1995, Watt 1995), with (Glassner 1995) providing a more in-depth look at image formation and rendering. For statistics and machine learning, Chris Bishop's (2006) book is a wonderful and comprehensive introduction with a wealth of exercises. Students may also want to look in other textbooks on computer vision for material that we do not cover here, as well as for additional project ideas (Ballard and Brown 1982, Faugeras 1993, Nalwa 1993, Trucco and Verri 1998, Forsyth and Ponce 2003).

There is, however, no substitute for reading the latest research literature, both for the latest ideas and techniques, and for the most up-to-date references to related literature.<sup>15</sup> In this book, I have attempted to cite the most recent work in each field so that students can read these directly and use them as inspiration for their own work. Browsing the last few years' conference proceedings from the major graphics and vision conferences such as SIGGRAPH, CVPR, ECCV, and ICCV will provide a wealth of new ideas. The tutorials offered at these conferences, for which slides and/or excellent notes are often available on-line, are also an invaluable resource.

---

<sup>15</sup> For a comprehensive bibliography and taxonomy of computer vision research, Keith Price's Annotated Computer Vision Bibliography <http://iris.usc.edu/Vision-Notes/bibliography/contents.html> is an invaluable resource.