

1

Introduction and background

Two of the important problems in computer vision are

- to locate, and determine the configuration of, a known object in a static image
- to locate, and determine the configuration of, a (moving, deforming) known object in each frame of a video sequence

The word “object” here could refer to a given physical object (the coffee mug in front of me), or to any element of a class of objects (apples, or cars). No general algorithm for solving these problems by computer is known. Indeed, some would claim that no such algorithm exists. To solve these problems satisfactorily, the “known object” must be defined, and in at least some cases this requires the computer to *understand* a general notion (e.g. “cars”) involving language. This in turn leads to a whole host of problems associated with language and consciousness which have been studied by philosophers, computer scientists and others working in artificial intelligence (e.g. Boden, 1990). Those who do not believe the human brain is equivalent to a Turing machine can plausibly appeal to our non-Turing-ness for an explanation as to why humans can solve the problems above with such apparent ease while computers perform so poorly. The rest of us must wait, think, and try to come up with better algorithms.

This book comprises a contribution to the waiting and thinking, while acknowledging that much more must be done before our computer vision algorithms can fit into the layman’s idea of the phrase “artificial intelligence”. A particular emphasis of the approach in this book is that statistics and probability underpin the algorithms and can be used to interpret the results. Each of the methods of localisation and tracking described later produces as output a probability distribution, which can be interpreted as the computer’s “belief” about the state of the world.

To summarise: we would like to use a computer to make inferences about the configurations of objects in images and video sequences. There are two

fundamental contributions to this objective in the book. The first is a coherent set of models which can be used to infer the location and configuration of objects based on simple properties (essentially their shape and colour), both in static images and as the objects move in video streams. These models, termed *contour likelihoods*, lead to statistical quantities such as likelihood ratios which can be easily interpreted. Refinements and extensions of the basic contour likelihood approach lead to models which are valid for partially occluded objects (the *Markov likelihood*), and to likelihoods which can be used for tracking more than one object simultaneously; the latter is an example of a “probabilistic exclusion principle”.

The second contribution is to the theory of particle filters, and their use in computer vision applications. The theory and basic results on particle filters are collected and proved in a new, self-contained formulation based on *particle sets*. The exposition is rigorous yet accessible to the vision community, and methods of assessing the efficacy of particle filters are also introduced. Finally, a technique called *partitioned sampling* is described. Partitioned sampling dramatically improves the performance of an important class of particle filters, and thereby provides a solution to certain tracking problems that were previously unassailable by particle filtering methods.

1.1 Overview

The diverse nature of the problems to be tackled means that a unified literature survey would tend to confuse rather than elucidate. Instead, literature surveys are included in each chapter where they are needed. In particular, Sections 2.7 and 2.8 are surveys on aspects of particle filtering, Section 4.1 surveys object localisation, Section 6.1 surveys multiple-target tracking and Chapter 3 begins with a brief overview of generative models.

The content of the book is as follows. The next section of this chapter provides an introduction to *active contours* — the particular set of models used to describe and make inferences on images throughout the book.

Chapter 2 (“The Condensation algorithm”) is a self-contained description of particle filters, which are known in the computer vision literature as the Condensation algorithm. Condensation is the algorithm used for all the tracking problems considered in the book, so Chapter 2 provides a proof of its asymptotic correctness and introduces concepts useful for assessing the effectiveness of particle filters.

The huge amount of data in images means that vision algorithms often consider only a much smaller set of data called *features*, such as edges or colour blobs, which are derived from the image in a prescribed manner. Statistical interpretation of the features then requires a *generative model* of the features — a statistical model describing how the features arise in typical images. Chapter 3 (“Contour likelihoods”) is a description of the generative models used in this book, and of the standard types of statistical inference that can be performed on these models.

Chapter 4 is a description of how to apply the contour likelihoods of Chapter 3 to real localisation and tracking problems. The methods are demonstrated on a variety of static localisation problems, and real-time tracking tasks.

The generative models of Chapter 3 assume that certain types of image features are statistically independent; this is an approximation that is violated badly when an object is partly occluded. Chapter 5 (“Modelling occlusions using the Markov likelihood”) is a self-contained solution to this problem which modifies the likelihoods by using a Markov random field.

Chapter 6 (“A probabilistic exclusion principle for multiple objects”) generalises the previous contour likelihoods so that they can be used with more than one object. It turns out that the correct way of doing this leads to a probabilistic exclusion principle: the computer’s hypotheses about the object configurations are prevented from coalescing unduly onto the same state.

As the final substantive chapter of the book, Chapter 7 (“Partitioned sampling”) introduces a method for dramatically improving the effectiveness of certain particle filters which must operate in high-dimensional spaces. Partitioned sampling is particularly suited to tracking multiple objects and articulated objects, and examples of both are demonstrated. A particular highlight is a real-time hand tracking system of sufficiently good quality that it can be used as the input for a drawing package.¹

Chapter 8, after some brief speculation on what might come next, directs the reader back here for a summary.

Some of the material in the book has appeared elsewhere, including in Blake et al. (1998), MacCormick and Blake (1998a, 1998b, 1999), MacCormick and Isard (2000), and Isard and MacCormick (2000). Video sequences and other material related to the book are available from the following web site:

<http://www.robots.ox.ac.uk/~jmac/visualtracking>

1.2 Active contours for visual tracking

The human visual system provides empirical proof that accurate tracking of moving, deforming objects is an information-processing problem with a robust, real-time solution. A full understanding of this (human) solution is still well beyond our grasp, and current solutions using *artificial* intelligence must make simplifying assumptions and accept less complete results to make progress. Several different paradigms for these simplifying assumptions are popular in the literature. If we accept that complete reconstruction of the “plenoptic function” (Adelson and Bergen, 1991) — or even more generally, a complete reconstruction of both the 3D structure of the world and its illuminance properties — is not a realistic goal, then we must instead adopt some basis for segmenting the world and our images of it into meaningful blocks. A simplistic taxonomy of approaches to this segmentation task might list the following: “layers” (e.g. Wang and Adelson, 1993; Baker et al., 1998), in which the world is taken to

¹This was developed in collaboration with Michael Isard.

consist of cardboard cutouts; “texture” (e.g. Chellappa and Chatterjee, 1985; Malik et al., 1999), in which the world consists of objects defined by homogeneous textures; and “contours” or “snakes” (e.g. Kass et al., 1987; Blake and Isard, 1998; Terzopoulos and Szeliski, 1992), in which the world consists of objects defined by boundaries with given properties.

We have to start somewhere to get somewhere else. So this book adopts the contour approach, and builds on a theory of shape and motion espoused by Andrew Blake and his co-workers at the University of Oxford. Termed “active contours”, this approach is described in detail in a book of the same name by Blake and Isard (1998). This section of the book will cover the bare bones of this theory (splines, shape spaces, and dynamical models using auto-regressive processes) for completeness, but the interested reader is urged to consult the *Active Contours* book, as well as research papers such as those by Blake et al. (1993, 1995), Curwen and Blake (1992), and Reynard et al. (1996) and, for a broader perspective, the *Active Vision* collection (Blake and Yuille, 1992). Important related work includes the Active Shape Models of Cootes, Taylor, and others in Manchester (e.g. Cootes et al., 1995), and the snakes of Kass et al. (1987). Some detailed applications of the active contour approach are given in Oxford Ph.D. theses by Wildenberg (1997), Kaucic (1997), Isard (1998), and North (1998).

1.2.1 Splines and shape space

Each tracking or localisation problem addressed by this book involves a distinguished class of objects called “targets” or “target objects”. Examples of typical classes of targets include cars, bicycles, humans, and hands. A given target object is described by its outline, which is modelled as a B-spline (Bartels et al., 1987). B-splines are a convenient way of representing smooth, natural-looking shapes by specifying only a small number of “control points”. Specifically, if the coordinates of the control points are $(x_1, y_1), \dots, (x_n, y_n)$ then the B-spline is a curve $(x(s), y(s))^T$ parameterised by a real variable s on an interval of the real line:

$$\begin{pmatrix} x(s) \\ y(s) \end{pmatrix} = B(s) \begin{pmatrix} \vec{x} \\ \vec{y} \end{pmatrix}, \quad (1.1)$$

where $B(s)$ is a $2 \times 2n$ matrix whose entries are polynomials in s , and $\vec{x}, \vec{y}$ are $n \times 1$ column vectors containing the x - and y -coordinates of the control points respectively. The key point to note is that for a given B , the space of all contours is finite-dimensional: indeed it is isomorphic to $\mathbb{R}^{2n}$, and can be given an inner product which makes the concept of “distance” between two splines meaningful (see Appendix A.1).

We will call any such B-spline a *contour*. Figure 1.1 gives an example of a mouse-shaped contour with seven control points. Typical objects require 5–20 control points to produce a contour that, for a human observer, appears to match the target object very closely. The vector space of such contours has 10–40 dimensions, which is often undesirably large. Hence we generally work in a vector subspace of $\mathbb{R}^{2n}$ termed the *shape space* and denoted by $\mathcal{X}$.

An element $\mathbf{x} \in \mathcal{X}$ is related to the control point coordinates $\vec{x}, \vec{y}$ by a linear transformation with a fixed offset:

$$\begin{pmatrix} \vec{x} \\ \vec{y} \end{pmatrix} = W\mathbf{x} + Q. \quad (1.2)$$

If the shape space has d dimensions, then W is the $2n \times d$ *shape matrix*, $\mathbf{x}$ is a $d \times 1$ column vector generally just referred to as the *configuration* or *state*, and Q is a $d \times 1$ vector called the *template* for this object.

Given a template for a rigid planar object, it is easy to define a shape space $\mathcal{X}$ corresponding to either translations, 2D Euclidean similarities or affine transformations² of the template. If the target object is articulated, non-rigid, or non-planar, it is still often possible to work in a linear space of low dimension by extending $\mathcal{X}$. This can be done on theoretical grounds, or by using keyframes from training data followed by principal components analysis (Jolliffe, 1986; Cootes et al., 1995). Actually, none of the object localisation techniques in this book require $\mathcal{X}$ to be a vector space. $\mathcal{X}$ can be any sub-manifold of the spline space and discussions will treat it as such.³ Even the tracking algorithms discussed later do not really require a linear shape space, although linearity is required to implement the dynamical models discussed in the next section.

1.2.2 Dynamical models using auto-regressive processes

Any tracking or filtering algorithm requires a model of how the system is expected to evolve over time. The computer vision community seems to divide into two camps here. On the one hand, one can attempt explicit physical realism by modelling kinematics and other forces (e.g. Robertson and Schwertassek, 1988; Bregler and Malik, 1998). The other chief approach is the *auto-regressive process* or ARP (Lutkepohl, 1993), which has been used in many speech and vision applications (e.g. Bobick and Wilson, 1995; Rabiner and Bing-Hwang, 1993; Black and Yacoob, 1995). This book adopts second-order ARPs exclusively. It has been our experience that these models capture a rich variety of motions of interest in computer vision tracking problems, while being both straightforward to learn from training data and easy to implement efficiently in real-time algorithms.

A second-order ARP expresses the state $\mathbf{x}_t$ at time t as a linear combination of the previous two states and some Gaussian noise:

$$\mathbf{x}_t = A_2\mathbf{x}_{t-2} + A_1\mathbf{x}_{t-1} + Bw,$$

where A_1, A_2, B are fixed $d \times d$ matrices and w is a $d \times 1$ vector of i.i.d. standard normal variates. Blake and Isard (1998) give some useful techniques for setting A_1, A_2, B by physical reasoning for relatively complex tracking problems, and

²For planar objects, these are equivalent to 3D Euclidean transformations with perspective effects neglected.

³Authors using contours seem to have restricted themselves to linear configuration spaces because of various computational advantages and because the Kalman filter requires a linear state space.

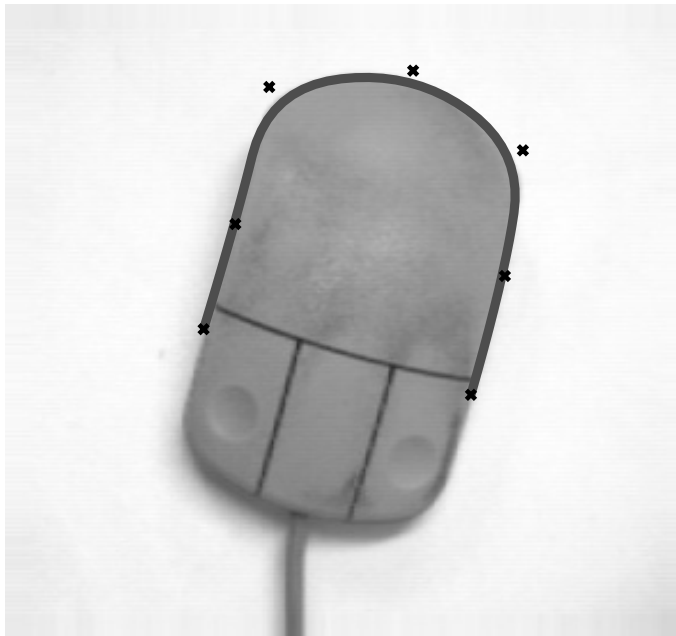


Figure 1.1: **A contour.** Throughout the book, a contour is a B-spline like this one, specified by control points shown here as black crosses. Note that the contour does not interpolate the control points.

then bootstrapping by taking maximum likelihood estimates based on longer and longer sequences of successful tracking. This is the approach used for all tracking in this book. The expectation-maximisation algorithm has also been used for learning ARPs; two computer vision applications can be found in North (1998), and Pavlovic et al. (1999).

1.2.3 Measurement methodology

To perform inferences on real data, one must not only acquire the data, but also possess a model of the data acquisition process. This section describes the process itself, while Chapter 3 addresses ways of modelling it. A plethora of tools have been used for this by the image analysis community, and we choose one based on its utility for tracking moving and deforming objects.

The method is demonstrated in Figure 1.2. At fixed⁴ points along the B-spline, line segments normal to the contour are cast onto the image. These normals are called *measurement lines*; typically they will be spaced equally around the contour, with between one and five measurement points per span of

⁴To be precise, the measurement points are not fixed on the contour at all — this concept makes no sense when the contour's shape can vary. The measurement points are actually placed at fixed values of the B-spline parameter s in equation 1.1.

the B-spline. Often the length of the measurement line is fixed in advance, and it is almost always assumed that the measurement line is symmetric about the measurement point. Next, a one-dimensional feature detector (a simple edge detector, for instance) is applied to the image intensity along each measurement line. For each line, the result is a list of numbers describing the distance from each feature found to the measurement point; these numbers are called the *feature innovations*.⁵

Chapter 3 describes some detailed models which can be used to perform inference on the distribution of the features which are detected by this method. Section 3.2.3 describes a different method of measuring the image, in which the measurement lines are fixed *in the image* rather than being attached to the contour.

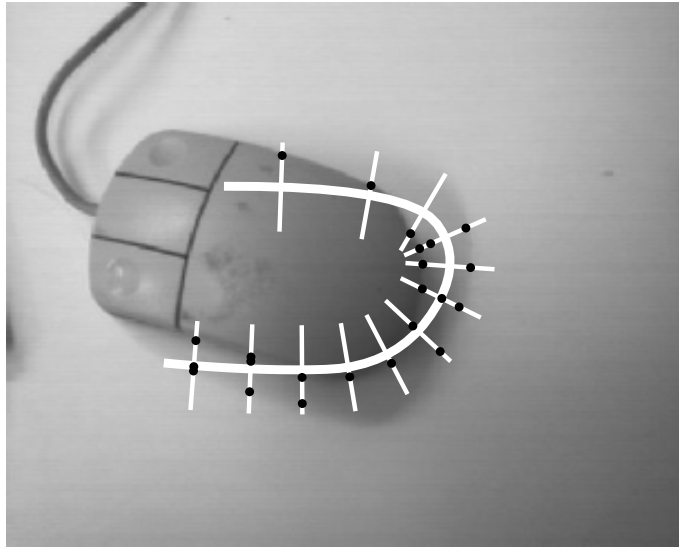


Figure 1.2: **Measurement methodology and notation.** The thick white line is $\mathbf{x}$ — a mouse-shaped contour in some hypothesised configuration. The thin lines are measurement lines, along which a one-dimensional feature detector is applied. Black dots show the output of the feature detector. The distance of a feature from the contour is called its innovation.

⁵The contour is assumed to have an orientation, and a negative feature innovation indicates the feature is in the interior of the contour. (By convention, the orientation is chosen so that the interior is on the left of an ant traversing the contour in the positive direction.) Note that both open and closed contours have an interior according to this definition.

Stochastic Algorithms for Visual Tracking
Probabilistic Modelling and Stochastic Algorithms for
Visual Localisation and Tracking

MacCormick, J.

2002, IX, 174 p., Hardcover

ISBN: 978-1-85233-601-1