massachusetts institute of technology — artificial intelligence laboratory

From First Contact to Close Encounters: A Developmentally Deep Perceptual System for a Humanoid Robot

Paul Fitzpatrick

AI Technical Report 2003-008 June 2003

From First Contact to Close Encounters: A Developmentally Deep Perceptual System for a Humanoid Robot

by

Paul Michael Fitzpatrick

B.Eng., University of Limerick (1995)
M.Eng., University of Limerick (1997)

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Doctor of Philosopy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2003

Certified by: Rodney A. Brooks
Fujitsu Professor of Computer Science and Engineering
Thesis Supervisor

Accepted by: Arthur C. Smith
Chairman, Department Committee on Graduate Students

From First Contact to Close Encounters:
A Developmentally Deep Perceptual System for a Humanoid Robot

by

Paul Michael Fitzpatrick

Submitted to the Department of Electrical Engineering and Computer Science on May 22, 2003, in partial fulfillment of the requirements for the degree of Doctor of Philosopy in Computer Science and Engineering

Abstract

This thesis presents a perceptual system for a humanoid robot that integrates abilities such as object localization and recognition with the deeper developmental machinery required to forge those competences out of raw physical experiences. It shows that a robotic platform can build up and maintain a system for object localization, segmentation, and recognition, starting from very little. What the robot starts with is a direct solution to achieving figure/ground separation: it simply 'pokes around' in a region of visual ambiguity and watches what happens. If the arm passes through an area, that area is recognized as free space. If the arm collides with an object, causing it to move, the robot can use that motion to segment the object from the background. Once the robot can acquire reliable segmented views of objects, it learns from them, and from then on recognizes and segments those objects without further contact. Both low-level and high-level visual features can also be learned in this way, and examples are presented for both: orientation detection and affordance recognition, respectively.

The motivation for this work is simple. Training on large corpora of annotated real-world data has proven crucial for creating robust solutions to perceptual problems such as speech recognition and face detection. But the powerful tools used during training of such systems are typically stripped away at deployment. Ideally they should remain, particularly for unstable tasks such as object detection, where the set of objects needed in a task tomorrow might be different from the set of objects needed today. The key limiting factor is access to training data, but as this thesis shows, that need not be a problem on a robotic platform that can actively probe its environment, and carry out experiments to resolve ambiguity. This work is an instance of a general approach to learning a new perceptual judgment: find special situations in which the perceptual judgment is easy and study these situations to find correlated features that can be observed more generally.

Thesis Supervisor: Rodney A. Brooks
Title: Fujitsu Professor of Computer Science and Engineering

Acknowledgments

My thanks to Rod Brooks for his support and advice throughout my time at MIT, and for running such a diverse and interesting research group – and letting me be a part of it. I am also grateful to Deb Roy and Trevor Darrell, the other members of my committee, for their helpful comments on the thesis.

The robots used in this thesis are the fruit of the combined labor of many students. I am grateful to them all, and particularly indebted to Brian Scassellati, Matthew Marjanović, Cynthia Breazeal, Matthew Williamson, and Bryan Adams.

I would like to take this opportunity to deny the existence of the Tea Conspiracy, a group which – if it existed – would no doubt have been a very pleasant break between working-all-day and working-all-night at the AI lab. Eduardo Torres-Jara strikes me as the sort who might ring-lead such a conspiracy, if it existed, which it emphatically does not (and even if it did, there is absolutely no evidence that its members are plotting a lab coup d'état). Thank you Eduardo for your indefatigable good-humor. My thanks to Giorgio Metta for being a great friend and colleague. To both Charlie and Clarke Kemp, for innumerable interesting and challenging conversations. To Artur Arsenio, Lijin Aryananda and Hideki Kozima, for making office 936 a fun place to be. And to everyone else in the Living Machines group – Martin Martin, Una-May O'Reilly, Paulina Varshavskaya, Aaron Edsinger, Jessica Banks, Jeff Weber, Jessica Howe, Annika Pfluger, and the mysterious Juan Velasquez.

My journey from herding goats in rural Ireland to doing robotics research in Cambridge was a long one. Boundless thanks to my parents, Ann and William, who made it all possible. And to my siblings, Deirdre, Liam, Mary, and John for their support. To Aunt Josephine and Granny F for writing to me so often while receiving so little in return. And to various groups in the governments in Ireland and the US for paying for my education – I might still be herding goats otherwise (not that there is anything wrong with that). I am grateful to the books of Terry Pratchett for providing endless amusement and inspiration. They are quoted at the start of each chapter of the thesis, and bear only the flimsiest of connection with the main text.

To Noémi Giszpenc for giving me the confidence that, if I could learn to pronounce her name, I could accomplish anything.

To my grandfathers,
Tom and Jerry.

Contents

List of Figures

CHAPTER 1

Introduction

> *Everything starts somewhere, although many physicists disagree. But people have always been dimly aware of the problems with the start of things. They wonder aloud how the snowplough driver gets to work, or how the makers of dictionaries look up the spellings of words.* (Pratchett, 1996)

The goal of this work is to build a perceptual system for a robot that integrates useful "mature" abilities, such as object localization and recognition, with the deeper developmental machinery required to forge those competences out of raw physical experiences. The motivation for doing so is simple. Training on large corpora of real-world data has proven crucial for creating robust solutions to perceptual problems such as speech recognition and face detection. But the powerful tools used during training of such systems are typically stripped away at deployment. For problems that are more or less stable over time, such as face detection in benign conditions, this is acceptable. But for problems where conditions or requirements can change, then the line between training and deployment cannot reasonably be drawn. The resources used during training should ideally remain available as a support structure surrounding and maintaining the current perceptual competences. There are barriers to doing this. In particular, annotated data is typically needed for training, and this is difficult to acquire online. But that is the challenge this thesis addresses. It will show that a robotic platform can build up and maintain a quite sophisticated object localization, segmentation, and recognition

system, starting from very little.

1.1 The place of perception in AI

If the human brain were a car, this message would be overlaid on all our mental reflections: “caution, perceptual judgements may be subtler then they appear”. Time and time again, the difficulty of implementing analogues of human perception has been underestimated by AI researchers. For example, the Summer Vision Project of 1966 at the MIT AI Lab apparently expected to implement figure/ground separation and object recognition on a limited set of objects such as balls and cylinders in the month of July, and then extend that to cigarette packs, batteries, tools and cups in August (Papert, 1966). That “blind spot” continues to the current day – for example, the proposal for the thesis you are reading blithely assumed the existence of perceptual abilities that now consume entire chapters. But there has been progress. Results in neuroscience continue to drive home the sophistication of the perceptual machinery in humans and other animals. Computer vision and speech recognition have become blossoming fields in their own right. Advances in consumer electronics have led to a growing drive towards advanced human/computer interfaces, which bring machine perception to the forefront. What does all this mean for AI, and its traditional focus on representation, search, planning, and plan execution? For devices that need to operate in rich, unconstrained environments, the emphasis on planning may have been premature:

> “I suspect that this field will exist only so long as it is considered acceptable to test these schemes without a realistic perceptual interface. Workers who have confronted perception have found that on the one hand it is a much harder problem than action selection and that on the other hand once it has been squarely faced most of the difficulties of action selection are eliminated because they arise from inadequate perceptual access in the first place.” (Chapman, 1990)

It is undeniable that planning and search are crucial for applications with complex logistics, such as shipping and chess. But for robotics in particular, simply projecting from the real world onto some form where planning and search can be applied seems to be the key research problem: “This abstraction process is the essence of intelligence and the hard part of the problem being solved” (Brooks, 1991b). Early approaches to machine perception in AI focused on building and maintaining detailed, integrated models of the world that were as complete as possible given the sensor data available. This proved extremely difficult, and over time more practical approaches were developed. Here are cartoon-caricatures of some of them:

- ▷ **Stay physical:** Stay as close to the raw sensor data as possible. In simple cases, it may be possible to use the world as its own model and avoid the difficulties involved in creating and maintaining a representation of a noisily- and partially-observed world (Brooks, 1991b). Tasks such as obstacle avoidance can be achieved reactively, and Connell (1989) gives a good example of how a task with temporal structure can be performed by maintaining state in the world and the robot's body rather than within its control system. This work clearly demonstrates that the structure of a task is logically distinct from the structures required to perform it. Activity that is sensitive to some external structure in the world does not imply a control system that directly mirrors that structure in its organization.
- ▷ **Stay focused:** Adopt a point of view from which to describe the world that is sufficient for your task and which simplifies the kind of references that need to be made, hopefully to the point where they can be easily and accurately maintained. Good examples include deictic representations like those used in Pengi (Chapman and Agre, 1987), or Toto's representations of space (Mataric, 1990).
- ▷ **Stay open:** Use multiple representations, and be flexible about switching between representations as each run into trouble (Minsky, 1985). This idea overlaps with the notion of encoding common sense (Lenat, 1995), and using multiple partial theories rather than searching – perhaps vainly – for single unified representations.

While there are some real conflicts in the various approaches that have been adopted, they also have a common thread of pragmatism running through them. Some ask "what is the minimal representation possible", others "what choice of representation will allow me to develop my system most rapidly?" (Lenat, 1995). They are also all steps away from an all-singing, all-dancing monolithic representation of the external world. Perhaps they can be summarized (no doubt kicking and screaming) with the motto "robustness from perspective" – if you look at a problem the right way, it may be relatively easy. This idea was present from the very beginning of AI, with the emphasis on finding the right representations for problems, but it seemed to get lost once division of labor set in and the problems (in some cases) got redefined to match the representations.

There is another approach to robust perception that has developed, and that can perhaps be described as "robustness from experience". Drawing on tools from machine learning, just about any module operating on sensor input can be improved. At a minimum, its performance can be characterized empirically, to determine when it can be relied upon and when it fails, so that its output can be appropriately weighed against other sources. The same pro-

cess can be applied at finer granularity to any parameters within the module that affect its performance in a traceable way. Taking statistical learning of this kind seriously leads to architectures that seem to contradict the above approaches, in that they derive benefit from representations that are as integrated as possible. For example, when training a speech recognition system, it is useful to be able to combine acoustic, phonological, language models so that optimization occurs over the largest scope possible (Mou and Zue, 2001).

The success of statistical, corpus-based methods suggests the following additional organizing principle to the ones already enunciated :-

- ▷ **Stay connected:** Statistical training creates an empirical connection between parameters in the system and experience in the world that leads to robustness. If we can *maintain* that connection as the environment changes, then we can maintain robustness. This will require integrating the tools typically used during training with the deployed system itself, and engineering opportunities to replace the role that annotation plays.

This thesis argues that robots must be given not just particular perceptual competences, but the tools to forge those competences out of raw physical experiences. Three important tools for extending a robot's perceptual abilities whose importance have been recognized individually are related and brought together. The first is active perception, where the robot employs motor action to reliably perceive properties of the world that it otherwise could not. The second is development, where experience is used to improve perception. The third is interpersonal influences, where the robot's percepts are guided by those of an external agent. Examples are given for object segmentation, object recognition, and orientation sensitivity; initial work on action understanding is also described.

1.2 Why use a robot?

The fact that vision can be aided by action has been noted by many researchers (Aloimonos et al., 1987; Bajcsy, 1988; Ballard, 1991; Gibson, 1977). Work in this area focuses almost uniformly on the advantages afforded by moving cameras. For example, Klarquist and Bovik (1998) use a pair of cameras mounted on a track to achieve precise stereoscopic vision. The track acts as a variable baseline, with the system *physically* interpolating between the case where the cameras are close – and therefore images from them are easy to put into correspondence – and the case where the cameras are separated by a large baseline – where the images are different enough for correspondences to be hard to make. Tracking correspondences from the first to the

Figure 1.1: Training data is worth its weight in ice cream in the speech recognition research community (certificate created by Kate Saenko).

second case allows accurate depth estimates to be made on a wider baseline than could otherwise be supported.

In this thesis, the work described in Chapter 3 extends the basic idea of action-aided vision to include simple manipulation, rather than just moving cameras. Just as conventional active vision provides alternate approaches to classic problems such as stereo vision and object tracking, the approach developed here addresses the classic problem of object segmentation, giving the visual system the power to recruit arm movements to probe physical connectivity. This thesis is a step towards visual monitoring of robot action, and specifically manipulation, for the purposes of correction. If the robot makes a clumsy grasp due to an object being incorrectly segmented by its visual system, and ends up just brushing against an object, then this thesis shows how to exploit that motion to correctly segment the object – which is exactly what the robot needs to get the grasp right the next time around. If an object is awkwardly shaped and tends to slip away if grasped in a certain manner, then the affordance recognition approach is what is needed to learn about this and combat it. The ability to learn from clumsy motion will be an important tool in any real, general-purpose manipulation system.

Certain elements of this thesis could be abstracted from the robotic implementation and used in a passive system, such as the object recognition module described in Chapter 5. A protocol could be developed to allow a hu-

man teacher to present an object to the system and have it enrolled for object recognition without requiring physical action on the robot's part. For example the work of Nayar et al. (1996) detects when the scene before a camera changes, triggering segmentation and object enrollment. However, it relies on a very constrained environment – a dark background with no clutter, and no extraneous environmental motion. Another approach that uses human-generated motion for segmentation – waving, pointing, etc. – is described in Arsenio et al. (2003). The SAIL robot (Weng et al., 2000a) can be presented with an object by placing the object in its gripper, which it then rotates 360° in depth, recording views as it goes. But all these protocols that do not admit of autonomous exploration necessarily limit the types of applications to which a robot can be applied. This thesis serves as a proof of concept that this limitation is not essential. Other researchers working on autonomous development are motivated by appeals to biology and software complexity (Weng et al., 2000b). The main argument added here is that autonomy is simply unavoidable if we wish to achieve maximum robustness. In the absence of perfect visual algorithms, it is crucial to be able to adapt to local conditions. This is particularly clear in the case of object recognition. If a robot moves from one locale to another, it will meet objects that it has never seen before. If it can autonomously adapt to these, then it will have a greater range of applicability. For example, imaging a robot asked to "clear out the junk in this basement." The degree of resourcefulness required to deal with awkwardly shaped and situated objects make this a very challenging task, and experimental manipulation would be a very helpful technology for it.

1.3 Replacing annotation

Suppose there is some property P of the environment whose value the robot cannot usually determine. Further suppose that in some very special situations, the robot *can* reliably determine the property. Then there is the potential for the robot to collect training data from such special situations, and learn other more robust ways to determine the property P. This process will be referred to as "developmental perception" in this thesis.

Active and interpersonal perception are identified as good sources of these "special situations" that allow the robot to temporarily reach beyond its current perceptual abilities, giving the opportunity for development to occur. Active perception refers to the use of motor action to simplify perception (Ballard, 1991), and has proven its worth many times in the history of robotics. It allows the robot to experience percepts that it (initially) could not without the motor action. Interpersonal perception refers to mechanisms whereby the robot's perceptual abilities can be influenced by those around it, such as a hu-

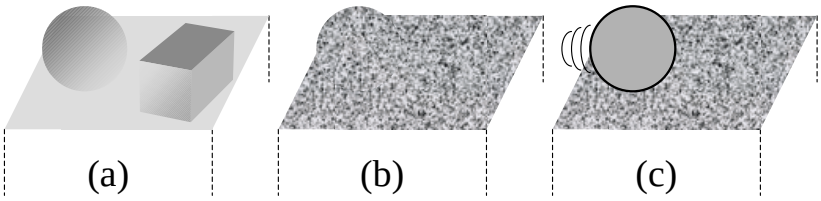


Figure 1.2: Cartoon motivation for active segmentation. Human vision is excellent at figure/ground separation (top left), but machine vision is not (center). Coherent motion is a powerful cue (right) and the robot can invoke it by simply reaching out and poking around.

man helper. For example, it may be necessary to correct category boundaries or communicate the structure of a complex activity.

By placing all of perception within a developmental framework, perceptual competence becomes the result of experience evoked by a set of behaviors and predispositions. If the machinery of development is sufficient to reliably lead to the perceptual competence in the first place, then it is likely to be able to regenerate it in somewhat changed circumstances, thus avoiding brittleness.

1.4 Active perception

The idea of using action to aid perception is the basis of the field of "active perception" in robotics and computer vision Ballard (1991); Sandini et al. (1993). The most well-known instance of active perception is active vision. The term "active vision" has become essentially synonymous with moving cameras, but it need not be. There is much to be gained by taking advantage of the fact that robots are actors in their environment, not simply passive observers. They have the opportunity to examine the world using causality, by performing probing actions and learning from the response. In conjunction with a developmental framework, this could allow the robot's experience to expand outward from its sensors into its environment, from its own arm to the objects it encounters, and from those objects both back to the robot itself and outwards to other actors that encounter those same objects.

Active vision work on the humanoid robot Cog is oriented towards opening up the potentially rich area of manipulation-aided vision, which is still largely unexplored. Object segmentation is an important first step. Chapter 3 develops the idea of *active segmentation*, where a robot is given a "poking"

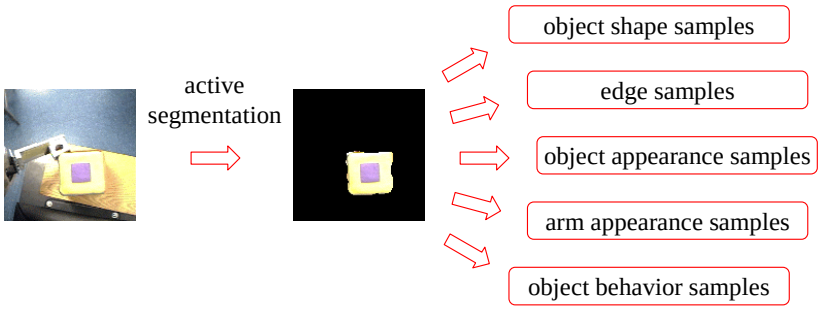


Figure 1.3: The benefits of active segmentation using poking. The robot can accumulate training data on the shape and appearance of objects. It can also locate the arm as it strikes objects, and record its appearance. At a lower level, the robot can sample edge fragments along the segmented boundaries and annotate them with their orientation, facilitating an empirical approach to orientation detection. Finally, tracking the motion of the object after poking is straightforward since there is a segmentation to initialize the tracker – hence the robot can record the motion that poking causes in different objects.

behavior that prompts it to select locations in its environment, and sweep through them with its arm. If an object is within the area swept, then the motion generated by the impact of the arm can be used to segment that object from its background, and obtaining a reasonable estimate of its boundary (see Figure 1.3). The image processing involved relies only on the ability to fixate the robot's gaze in the direction of its arm. This coordination can be achieved either as a hard-wired primitive or through learning. Within this context, it is possible to collect good views of the objects the robot pokes, and the robot's own arm. Giving the robot this behavior has several benefits. *(i)* The motion generated by the impact of the arm with an object greatly simplifies segmenting that object from its background, and obtaining a reasonable estimate of its boundary. This will prove to be key to automatically acquiring training data of sufficient quality to support the forms of learning described in the remainder of this thesis. *(ii)* The poking activity also leads to object-specific consequences, since different objects respond to poking in different ways. For example, a toy car will tend to roll forward, while a bottle will roll along its side. *(iii)* The basic operation involved, striking objects, can be performed by either the robot or its human companion, creating a controlled point of comparison between robot and human action.

Figure/ground separation is a long-standing problem in computer vision, due to the fundamental ambiguities involved in interpreting the 2D projection of a 3D world. No matter how good a passive system is at segmentation, there

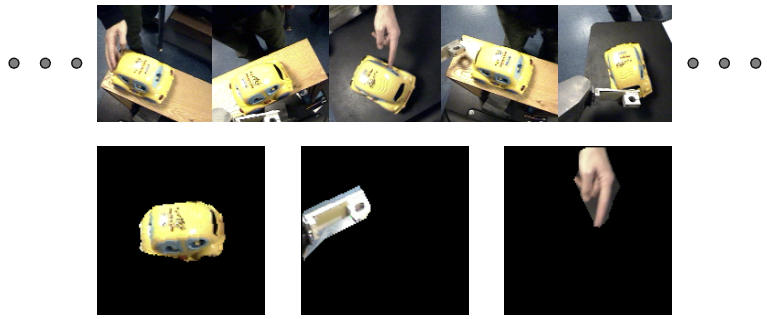


Figure 1.4: The top row shows sample views of a toy car that the robot sees during poking. Many such views are collected and segmented. The views are aligned to give an average prototype for the car (and the robot arm and human hand that acts upon it). To give a sense of the quality of the data, the bottom row shows the segmented views that are the best match with these prototypes. The car, the robot arm, and the hand belong to fundamentally different categories. The robot arm and human hand cause movement (are actors), the car suffers movement (is an object), and the arm is under the robot's control (is part of the self).

will be times when only an active approach will work, since visual appearance can be arbitrarily deceptive. Of course, there will be plenty of limitations on active segmentation as well. Segmentation through poking will not work on objects the robot cannot move, either because they are too small or too large. This is a constraint, but it means we are well matched to the space of manipulable objects, which is an important class for robotics.

1.5 Developmental perception

Active segmentation provides a special situation in which the robot can observe the boundary of an object. Outside of this situation, locating the object boundary is basically guesswork. This is precisely the kind of situation that a developmental framework could exploit. The simplest use of this information is to empirically characterize the appearance of boundaries and oriented visual features in general. Once an object boundary is known, the appearance of the edge between the object and the background can be sampled along it, and labelled with the orientation of the boundary in their neighborhood.

This is the subject of Chapter 4. At a higher-level, the segmented views provided by poking objects can be collected and clustered as shown in Figure 1.4. Such views are just what is needed to train an object detection and recognition system, which will allow the robot to locate objects in other, non-poking contexts. Developing object localization and recognition is the topic of Chapter 5.

Poking moves us one step outwards on a causal chain away from the robot and into the world, and gives a simple experimental procedure for segmenting objects. One way to extend this chain out further is to try to extract useful information from seeing a familiar object manipulated by someone else. This offers another opportunity for development – in this case, learning about other manipulators. Locating manipulators is covered in Chapter 6.

Another opportunity that poking provides is to learn how objects move when struck – both in general, for all objects, and for specific objects such as cars or bottles that tend to roll in particular directions. Given this information, the robot can strike an object in the direction it tends to move most, hence getting the strongest response and essentially evoking the "rolling affordance" offered by these objects. This is the subject of Chapter 7.

1.6 Interpersonal perception

Perception is not a completely objective process; there are choices to be made. For example, whether two objects are judged to be the same depends on which of their many features are considered essential and which are considered incidental. For a robot to be useful, it should draw the same distinctions a human would for a given task. To achieve this, there must be mechanisms that allow the robot's perceptual judgements to be channeled and moulded by a caregiver. This is also useful in situations where the robot's own abilities are simply not up to the challenge, and need a helping hand. This thesis identifies three channels that are particularly accessible sources of shared state: space, speech, and task structure. Robot and human both inhabit the same space. Both can observe the state of their workspace, and both can manipulate it, although not to equal extents. Chapter 8 covers a set of techniques for observing and maintaining spatial state. Another useful channel for communicating state is speech, covered in Chapter 9. Finally, the temporal structure of states and state transitions is the topic of Chapter 10.

1.7 Roadmap

Chapter 2	Overview of robot platforms and computational architecture
Chapter 3	Active segmentation of objects using poking
Chapter 4	Learning the appearance of oriented features
Chapter 5	Learning the appearance of objects
Chapter 6	Learning the appearance of manipulators
Chapter 7	Exploring an object affordance
Chapter 8	Spatially organized knowledge
Chapter 9	Recognizing and responding to words
Chapter 10	Interpersonal perception and task structure
Chapter 11	Discussion and conclusions

CHAPTER 2

The campaign for real time: robot bodies and brains

> *Wobbler had written an actual computer game like this once. It was called "Journey to Alpha Centauri." It was a screen with some dots on it. Because, he said, it happened in real time, which no-one had ever heard of until computers. He'd seen on TV that it took three thousand years to get to Alpha Centauri. He had written it so that if anyone kept their computer on for three thousand years, they'd be rewarded by a little dot appearing in the middle of the screen, and then a message saying, "Welcome to Alpha Centauri. Now go home."* (Pratchett, 1992a)

This work was implemented on two robots, Cog and Kismet (see Figure 2.1), developed at the Humanoid Robotics Group at the MIT AI Lab by various students over the past decade. More accurately, it was implemented on their sprawling "brains" – racks of computers connected to the bodies by a maze of cables, an extravagant sacrifice offered up to the gods of real-time performance. This chapter dips into the minimum detail of these systems necessary to understand the rest of the thesis. The interested reader is referred to the excellent theses of Williamson (1999), Breazeal (2000), and Scassellati (2001), on whose shoulders the author stands (or is at least trying to peer over).

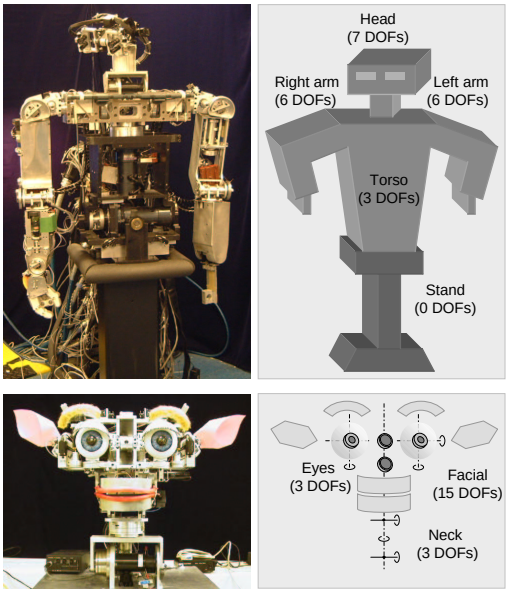


Figure 2.1: The robots Cog (top) and Kismet (bottom). Kismet is an expressive anthropomorphic head useful for human interaction work; Cog is an upper torso humanoid more adept at object interaction.

2.1 Cog, the strong silent type

Cog is an upper torso humanoid that has previously been given abilities such as visually-guided pointing (Marjanović et al., 1996), rhythmic operations such as turning a crank or driving a slinky (Williamson, 1998a), and responding to some simple forms of joint attention (Scassellati, 2000). For a good overview of the research agenda that Cog embodies, see Brooks et al. (1999).

2.1.1 Low-level arm control

Cog has two arms, each of which has six degrees of freedom organized as shown in Figure 2.2. The joints are driven by series elastic actuators (Williamson, 1995) – essentially a motor connected to its load via a spring (think strong and torsional rather than loosely coiled). The arm is not designed to enact trajectories with high fidelity. For that a very stiff arm is preferable. Rather, it is designed to perform well when interacting with a poorly characterized environment. The spring acts as a low pass filter for the friction and backlash effects introduced by gears, and protects the gear teeth

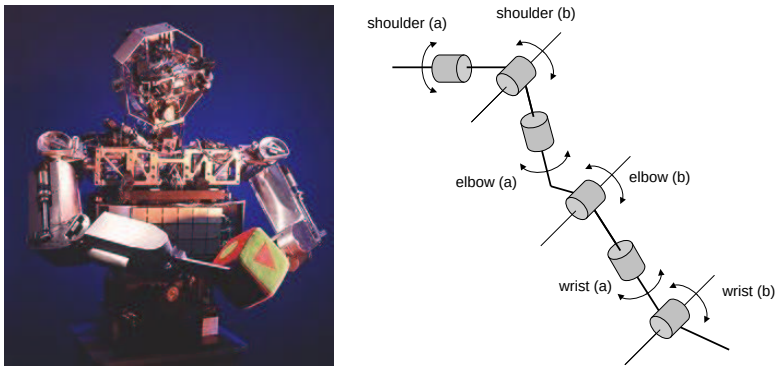


Figure 2.2: Kinematics of the arm, following Williamson (1999). There are a total of six joints, divided into a pair for each of the shoulder, elbow, and wrist/flipper.

from shearing under the impact of shock loads. A drawback to the use of series elastic actuators is that they limit the control bandwidth in cases where the applied force needs to change rapidly. The force applied by an electric motor can normally be changed rapidly, since it is directly proportional to the current supplied. By putting a motor in series with a spring, this ability is lost, since the motor must now drive a displacement of the spring's mass before the applied force changes. For the robot's head, which under normal operation should never come into contact with the environment, and which needs to move continuously and rapidly, series elastic actuators were not used. But for the arms, the tradeoff between control bandwidth and safety is appropriate. Robot arms are usually employed for the purposes of manipulation, but for this work they instead serve primarily as aides to the visual system. The target of a reaching operation is not assumed to be well characterized; in fact the reaching operation serves to better define the characteristics of the target through active segmentation (see Chapter 3). Hence the arm will habitually be colliding with objects. Sometimes the collisions will be with rigid, more or less unyielding structures such as a table. Sometimes the collisions will be with movable objects the robot could potentially manipulate. And sometimes the collisions will be with people. So it is important that both the physical nature of the arms, and the manner in which they are controlled, be tolerant of "obstacles".

The arms are driven by two nested controllers. The first implements force control, driving each motor until a desired deflection of the associated spring is achieved, as measured by a strain gauge. This high-speech control loop is implemented using an 8-axis motor controller from Motion Engineering,

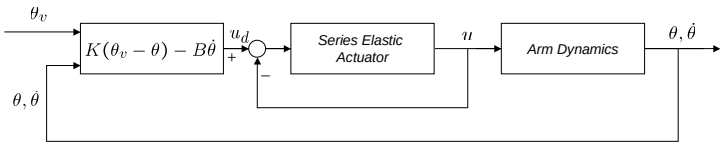


Figure 2.3: Control of a joint in the arm, following Williamson (1999). An inner loop controls the series elastic actuator in terms of force, working to achieve a desired deflection of the spring as measured by a strain gauge. An outer loop controls the deflection setpoint to achieve a desired joint angle, as measured by a potentiometer.

Inc. A second loop controls the deflection setpoint to achieve a desired joint angle as measured by a potentiometer. Figure 2.3 shows this second loop, following (Williamson, 1999). Various extensions and modifications to this basic approach have been made, for example to incorporate a feed-forward gravity compensating term, but the details are beyond the scope of this thesis.

2.1.2 Low-level head control

Figure 2.4 shows the degrees of freedom associated with Cog's head. In each "eye", a pair of cameras with different fields of view provides a step-wise approximation to the smoothly varying resolution of the human fovea (Scassellati, 1998). The eyes pan independently and tilt together. The head rolls and tilts through a differential drive. There is a further pan and tilt associated with the neck. There are a number of redundancies in the degrees of freedom to permit rapid movement of the eyes followed by a slower compensating motion of the relatively massive head. The head contains a 3-axis inertial sensor to simplify gaze stabilization.

The motors of the head are connected to optical encoders and driven by an 8-axis motor controller from Motion Engineering, Inc. The motor controller is configured to permit both position and velocity control. Much has been written about both the low-level and strategic control of such a head – see, for example Scassellati (2001) – so the details will be omitted here.

2.2 Kismet, the cute one

Parts of this work were developed on and reported for Kismet. Kismet is an "infant-like" robot whose form and behavior is designed to elicit nurturing responses from humans (Breazeal et al., 2001). It is essentially an active vision head augmented with expressive facial features so that it can both send and

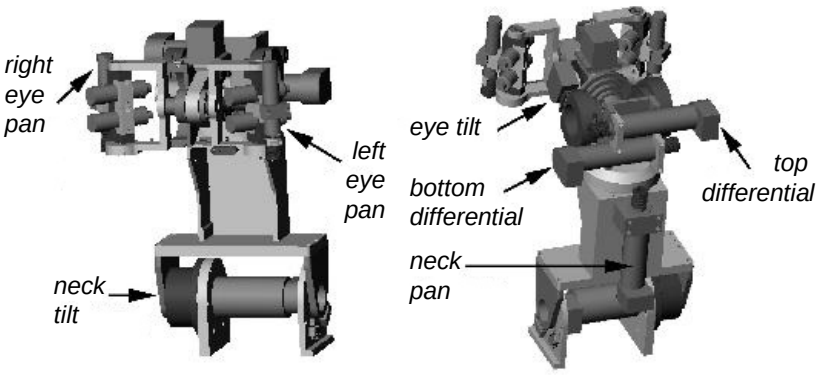


Figure 2.4: The motors of Cog's head, following Scassellati (2001). The degrees of freedom are loosely organized as pertaining to either the eyes, head, or neck. Pan and tilt (but not roll) of the eyes can be achieved at high speed without moving the mass of the head.

Figure 2.5: Kismet, the cute one.

receive human-like social cues. Kismet has a large set of expressive features - eyelids, eyebrows, ears, jaw, lips, neck and eye orientation. The schematic in Figure 2.1 shows the degrees of freedom relevant to visual perception (omitting the eyelids!). The eyes can turn independently along the horizontal (pan), but turn together along the vertical (tilt). The neck can turn the whole head horizontally and vertically, and can also crane forward. Two cameras with narrow fields of view rotate with the eyes. Two central cameras with wide fields of view rotate with the neck. These cameras are unaffected by the orientation of the eyes.

The reason for this mixture of cameras is that typical visual tasks require both high acuity and a wide field of view. High acuity is needed for recog-

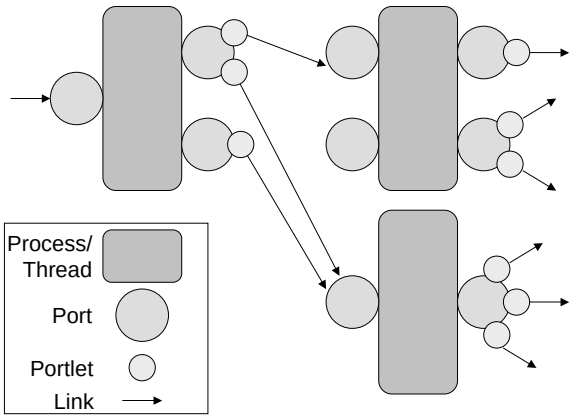


Figure 2.6: Communications model. Every process or thread can own any number of ports. Every port can be directed to send data to any number of other ports. Since different processes will access this data at different rates, it is useful to consider each port as owning several "portlets" that manage each individual link to another port. Given the most conservative quality of service settings, data will persist in the communications system as long as is necessary to send it on the slowest link.

nition tasks and for controlling precise visually guided motor movements. A wide field of view is needed for search tasks, for tracking multiple objects, compensating for involuntary ego-motion, etc. A common trade-off found in biological systems is to sample part of the visual field at a high enough resolution to support the first set of tasks, and to sample the rest of the field at an adequate level to support the second set. This is seen in animals with foveate vision, such as humans, where the density of photoreceptors is highest at the center and falls off dramatically towards the periphery. This can be implemented by using specially designed imaging hardware, space-variant image sampling (Schwartz et al., 1995), or by using multiple cameras with different fields of view, as we have done.

2.3 The cluster

Cog is controlled by a network of 32 computers, with mostly 800 MHz processors. Kismet is controlled by a similar but smaller network and four Motorola 68332 processors. The network was designed with the demands of real-time vision in mind; clearly if it was acceptable to run more slowly (say, one update a second instead of thirty) then a single machine could be used.

The primary engineering challenge is efficient interprocess communication across the computer nodes. We chose to meet this challenge by using QNX, a real-time operating system with a very clean, transparent message-passing system. On top of this was build an abstraction to support streaming communications and modular, subsumption-like design.

2.4 Cluster communication

Any process or thread can create a set of Ports. Ports are capable of communicating with each other, and shield the complexity of that communication from their owner. As far as a client process or thread is concerned, a Port is a fairly simple object. The client assigns a name to the Port, which gets registered in a global namespace. The client can hand the Port a piece of data to transmit, or read data the Port has received either by polling, blocking, or callback. There are some subtleties in the type of service required. The client can specify the kind of service required – a sender can specify whether transmission be guaranteed, or whether new data override data not yet sent; a receiver can independently specify whether, of the data the sender attempts to pass on, reception should be guaranteed or whether new data received by a Port should override data not yet read by its owner, and under what conditions the sending Port should be made to wait for the receiver.

Objects passed to the communications system obey a Pool interface. They can be cloned and recycled. The Port will clone objects as necessary, with an associated Pool growing to whatever size is required. This will depend on the rates at which all the links attached to the Port (via Portlets) read data at. By default, the owner of the Port is insulated from needing to know about that. For simple objects, cloning can be achieved with simple copies. The complex objects, such as images, a reference counting approach is worth using. Overall, this approach avoids unnecessary copies, and minimizes the allocation/deallocation of objects in the communications system. It is compatible with the existence of "special" memory areas managed by other entities, such as a framegrabber.

Ports and Portlets either use native QNX messaging for transport, or sockets if running on or communicating with a non-QNX system. The name server used is QNX's native `nameloc` service, or a simple socket-based `wide_nameloc` service for communicating with a non-QNX system. By default, these issues are transparent to client code. The system can operate transparently alongside other methods of communication, since it doesn't require any special resources such as control of the main process.

CHAPTER 3

First contact: tapping into the world

> *The Bursar shrugged.*
> *"This pot," he said, peering closely, "is actually quite an old Ming vase."*
> *He waited expectantly.*
> *"Why's it called Ming?" said the Archchancellor, on cue.*
> *The Bursar tapped the pot. It went *ming*.* (Pratchett, 1990)

Vision and action are intertwined at a very basic level in humans (Iacoboni et al., 1999). Researchers in machine vision have found many pragmatic reasons for integrating sensing tightly with motor control on an active vision head. This chapter extends this logic to simple object manipulation, showing how a simple tapping/poking behavior can help figure/ground separation. Poking an object makes it move, and motion is a powerful cue for visual segmentation. Poking itself does not require that an object be accurately segmented, since it can be performed simply as a sweep of the arm through a general neighborhood The periods immediately before and after the moment of impact turn out to be particularly informative, and give visual evidence for the boundary of the object that is well suited to segmentation using graph cuts. Of course, an experienced adult can interpret visual scenes perfectly well without acting upon them, and ideally our robots should do the same. Poking is proposed as a fallback segmentation method when all else fails, and a developmental opportunity for training up a contact-free object segmentation module. This topic is elaborated in later chapters.

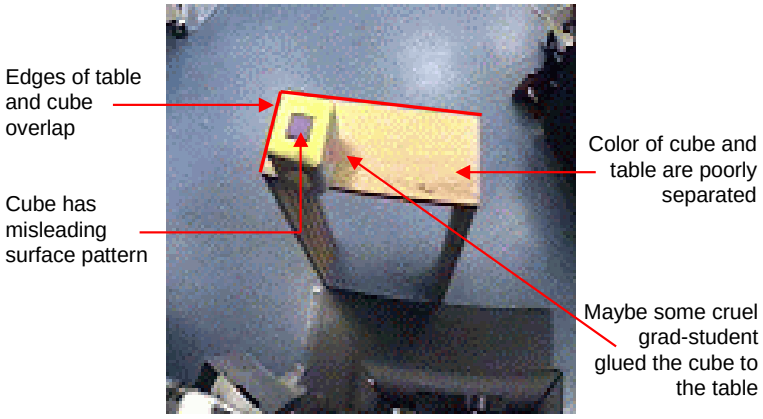


Figure 3.1: A cube on a table, to illustrate some problems in segmentation. The edges of the table and cube happen to be aligned, the colors of the cube and table are not well separated, and the cube has a potentially confusing surface pattern. And even if dense 3D information were available, there is no way to be really be sure the cube is an independently manipulable entity, and not connected to the table below it.

One contribution of this work is to clearly formulate a new object segmentation challenge not yet attended to in the machine vision literature, but which will become increasingly important in robotics. That challenge is: how can segmentation best be performed if exploratory manipulation is permissible? Another contribution is to demonstrate an approach that makes use of the most rudimentary manipulation possible to achieve segmentation, establishing a qualitative lower bound on what is possible, and showing that the benefits are non-trivial. Of course, segmentation is just the start of what can ultimately be learned through manipulation – but it is a good start, and given the complexity of dextrous manipulation, it is encouraging that even very simple motor control can lead to worthwhile results.

3.1 Active vision

A vision system is said to be *active* if it is embedded within a platform that can change its physical configuration to improve perceptual performance. For example, a robot's cameras might servo a rapidly moving target in order to stabilize the image and keep the target in view. The term is also used when processing is adapted to the current situation. Historically, a number of log-

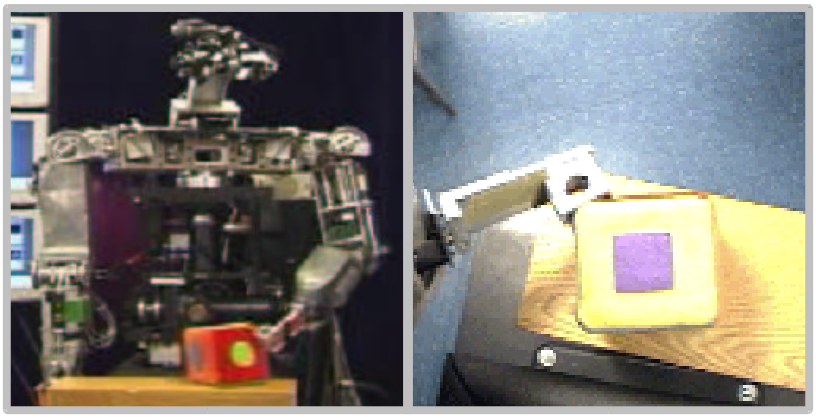


Figure 3.2: Resolving visual ambiguity by active means. The robot (left) reaches towards an object in its environment while fixating it with a camera. The robot's view is shown on the right. The boundary between the cube and the table it is sitting on is clear to human eyes, but too subtle to be reliably segmented by current automatic methods. But once the robot arm comes in contact with the object, it can be easily segmented from the background using the motion due to the impact.

ically distinct ideas are often associated with active vision. The first is that vision should be approached within the the context of an overall task or purpose (Aloimonos et al., 1987). Another idea is that if an observer can engage in controlled motion, it can integrate visual data from moment to moment to solve problems that are ill-posed statically. Well-chosen motion can simplify the computation required for widely studied vision problems, such as stereo matching (Bajcsy, 1988; Ballard, 1991). These interwoven ideas about active vision are teased apart in Tarr and Black (1994).

This work seeks to add two new threads to the mix. The first is that although active vision is often equated with moving cameras, the entire body of a robot could potentially be recruited to cooperate with the vision system. In this chapter, movement of the robot's arm is recruited to augment its visual system, and in particular to solve the figure/ground separation problem by active means. The second thread is that active systems have the potential to perform experiments that get close to accessing physical ground truth, and so potentially admit of a perceptual system that develops autonomously. The poking behavior gives the robot access to high quality training data that can be used to support object localization, segmentation, and recognition. It also

provides a simpler "fall-back" mechanism for segmentation, so that the robot is not entirely at the mercy of the failures of these higher-level competences.

3.2 Manipulation-driven vision

There are at least three clear situations in which it is useful for manipulation to guide vision, rather than the other way around, as is typical in robotics :-

- ▷ **Experimentation:** Making progress when perception is ambiguous.
- ▷ **Correction:** Recovering when perception is misleading.
- ▷ **Development:** Bootstrapping when perception is dumb.

Experimentation

Rather than simply failing in visually ambiguous situations, an active robotic platform has the potential to perform experiments on its environment that resolve the ambiguity. Consider the example in Figure 3.1. Visually, there are difficulties with segmenting this scene, due to some unfortunate coincidences in the alignment and color of the cube and the table. Rather than simply giving up on such situations, we could instead simply dispatch the robot's arm to the ambiguous region and poke around a bit. Several methods for characterizing the shape of an object through tactile information have been developed, such as shape from probing (Cole and Yap, 1987; Paulos, 1999) or pushing (Jia and Erdmann, 1998; Moll and Erdmann, 2001). The work in this chapter exploits the fact that the *visual* feedback generated when the robot moves an object is highly informative, even when the motion is short and poorly controlled, or even accidental. The vocabulary used to describe the robot's motion – "tapping" or poking" as opposed to "probing" – is deliberately chosen to convey the idea of a quick jab (to evoke visual data) instead of an extended grope (for tactile data). Although tactile and visual information could usefully be combined, no tactile or proprioceptive information is assumed in this chapter – not even to determine whether the robot is in contact with an object.

Recovery

How a robot should grasp an object depends on its size and shape. Such parameters can be estimated visually, but this is bound to be fallible – particularly for unrecognized, unfamiliar objects. Failure may result in a clumsy grasp or glancing blow against the object. If the robot does not learn something from the encounter, then it will be apt to repeat the same mistake again

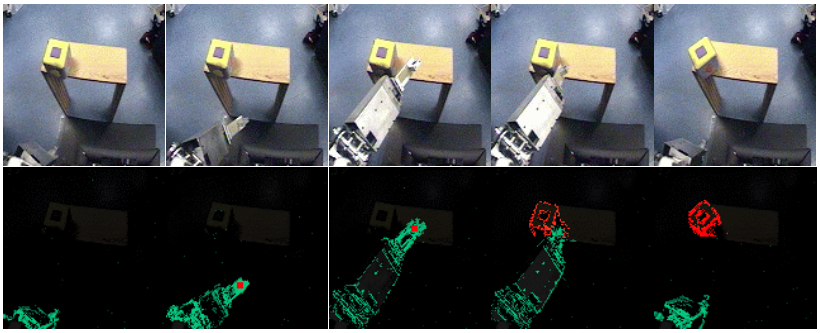


Figure 3.3: The upper sequence shows an arm extending into a workspace, tapping an object, and retracting. This is an exploratory mechanism for finding the boundaries of objects, and essentially requires the arm to collide with objects under normal operation, rather than as an occasional accident. The lower sequence shows the shape identified from the tap using simple image differencing and flipper tracking.

and again. As already foreshadowed, this chapter shows how to recover information about an object's extent by poking it, either accidentally or deliberately. This opens the door to extracting information from failed actions such as a glancing blow to an object during an attempt at manipulation, giving the robot the data it needs to do better next time.

Development

It would be cumbersome to always have to poke around to segment an object each time it comes into view. But the cleanly segmented views of objects generated by poking are exactly what is needed to train up an object recognition system, which in turn makes contact-free segmentation possible. So the kind of active segmentation proposed here can serve as an online teacher for passive segmentation techniques. Analogously, while an experienced adult can interpret visual scenes perfectly well without acting upon them, linking action and perception seems crucial to the developmental process that leads to that competence (Fitzpatrick and Metta, 2002).

3.3 Implementing active segmentation

Figure 3.3 shows frames from an early experiment where a robot arm was driven through a pre-programmed open-loop trajectory that swept through an area above a table, watched by a fixed camera. In the course of this trajectory, the arm came into contact with a cube sitting on a table, and disturbs it. The challenge is to identify and isolate this disturbance, and to use it to segment the cube, or whatever object the arm might encounter. The video stream from this and similar experiments were used to develop a baseline implementation of active segmentation and to clarify the requirements in terms of processing and behavior. The remainder of this chapter then fits this work into an actual behaving robot.

A reasonable way to segment the object would be to track the motion of the arm as it swings outwards, and to look for any motion that is not plausibly associated with the arm itself, but nevertheless appears to be physically adjacent to it. For this simple motivating example, the end-effector (or "flipper") is localized as the arm sweeps rapidly outwards using the heuristic that it lies at the highest point of the region of optic flow swept out by the arm in the image. The reaching trajectory of the robot relative to the camera orientation is controlled so that this is true. The sweeping motion is also made rather gentle, to minimize the opportunity for the motion of the arm itself to cause confusion. The motion of the flipper is bounded around the endpoint whose location we know from tracking during the extension phase, and can be subtracted easily. Flow not connected to the end-effector can be ignored as a distractor.

The sequence shown in Figure 3.3 is about the simplest case possible for segmenting the motion of the object. In practice, we would rather have less constraints on the motion of the arm, so we can approach the object from any convenient direction. It is also desirable to be able to explore areas where an object is likely to be, rather than simply sweeping blindly. But if objects are not already segmented, where can a target for poking come from? This is in fact very straightforward. As described in Section 2, Cog has an attentional system that allows it to locate and track salient visual stimuli. This is based entirely on low-level features such as color, motion, and binocular disparity that are well defined on small patches of the image, as opposed to features such as shape, size, and pose which only make sense on well-segmented objects. If Cog's attention system locates a patch of the image that seems reachable (based on disparity and overall robot pose) that is all it needs to know to reach toward it and attempt to poke it so that it can determine the physical extent of the object to which that patch belongs. A human can easily encourage this behavior by bringing an object close to the robot, moving it until the robot fixates it, and then leaving it down on the table. The robot will

track the object down to the table (without the need or the ability to actually segment it), observe that it can be reached, and poke it. Still, there are many things that could go wrong. Here is a list of the many potential pitfalls that could result in an inaccurate segmentation :-

- ▷ Object motion may not be particularly visible – if it is not highly textured, then there may be regions on its projection where optic flow is low or non-existent.
- ▷ The motion of the manipulator might be incorrectly separated from the motion of an object it comes in contact with.
- ▷ Unrelated motion in the background might be mistaken for movement of the manipulator or an impacted object.
- ▷ The manipulator might fail to come into contact with any object.
- ▷ The manipulator might obscure the robot's view of an object as it hits it.
- ▷ The object might not move rigidly, or might move too little or too much to be processed well.
- ▷ Motion of the camera might be mistaken for movement of the manipulator or an impacted object.

The first three points are dealt with by using a segmentation method based on graph cuts that allows diverse local evidence to be factored into making a good approximation to a globally optimal segmentation. Optic flow in textured regions provides evidence of movement, lack of optic flow in textured regions suggests lack of motion, comparing motion before and after the moment of impact gives evidence for what part of the image is the manipulator. The next three points are dealt with by careful engineering of the kinds of motion the robot makes. The final point is dealt with simply by keeping the camera fixated during poking. There is no real advantage to moving it, and segmentation based on motion viewed by a fixed camera is well understood and has been explored exhaustively (see for example Ross (2000); Stauffer (1999)).

3.4 First contact

If the object is to be segmented based on motion, we need to differentiate its motion from any other sources in the scene – particularly that of the robot itself. A high-quality opportunity to do this arises right at the moment of first contact between the robot and the object. This contact could be detected from tactile information, but it is also straightforward to detect visually, which is the method described here. The advantage of using visual information is that the same techniques can be applied to contact events about which the robot

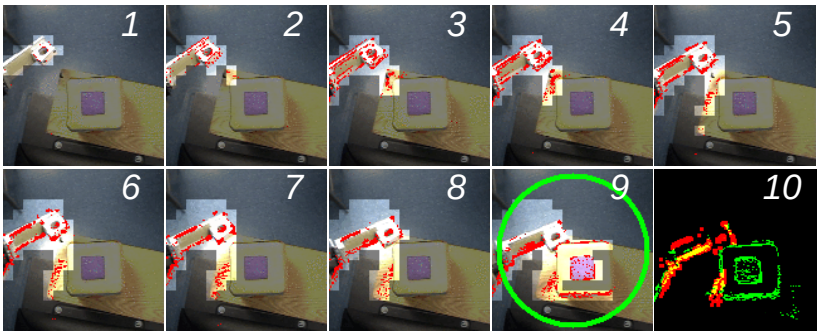


Figure 3.4: The moment of (ground) truth – detecting the point of impact between the robot's arm and an object. As the arm swings in, its motion is tracked frame by frame and aggregated within relatively low-resolution bins (highlighted squares). When an implausibly large spread in motion is detected across these bins, higher resolution processing is activated and segmentation begins.

has no privileged knowledge, such as a human hand poking an object (see Section 6).

For real-time operation, the moment of contact is first detected using low-resolution processing, and then the images before and after the contact are subjected to more detailed (and slower) analysis as described in the following section. Figure 3.4 shows a visualization of the procedure used. When the robot is attempting to poke a target, it suppresses camera movement and keeps the target fixated for maximum sensitivity to motion. A simple Gaussian model is maintained for the (R, G, B) color values of each pixel, based on their value over the last ten frames (one third of a second) received. Significant changes in pixel values from frame to frame are detected and flagged as possible motion. As the arm moves in the scene, its motion is tracked and discounted, along with its shadow and any background motion. Any area that the arm moves through is marked as "clear" of the object for a brief period – but not permanently since the arm may cross over the object before swinging back to strike it. An impact event is detected through a signature explosion of movement that is connected with the arm but spread across a much wider distance than the arm could reasonably have moved in the time available. Since the object is stationary before the robot pokes it, we can expect the variance of the Gaussians associated with the individual pixel models to be low. Hence they will be very sensitive to the pixel value changes associated with the sudden motion of the object. Once the impact is detected, we can drop briefly

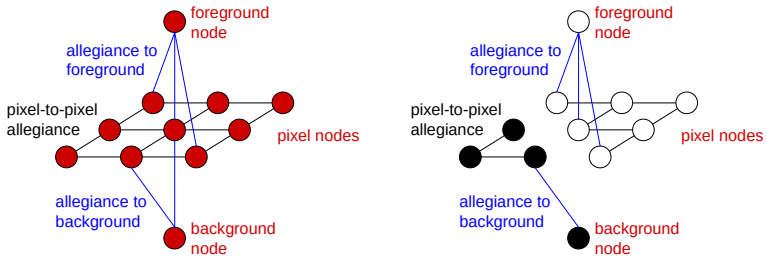


Figure 3.5: For a two-label problem on a 2D image, the input to a minimum-cut algorithm is typically as shown on the left. There is a node for each pixel, and two special nodes corresponding to the labels (foreground/background). Visual evidence is encoded on edges between the nodes. The output of the algorithm is shown on the right. The graph is cut into two disjoint sets, each containing exactly one of the special nodes, such that the total cost of the edges cut is (approximately) minimized.

out of real-time operation for a few seconds and perform the detailed analysis required to actually cleanly segment the object based on the apparent motion.

3.5 Figure/ground separation

Once the moment of contact is known, the motion visible before contact can be compared with the motion visible after contact to isolate the motion due to the object. Since we observe pixel variation rather than true motion, we can also factor in how we expect them to relate – for example, a highly textured region with no observed change over time can be confidently declared to be stationary, while a homogeneous region may well be in motion even if there is little observed change. In general, the information we have is sparse in the image and can be framed as probabilities that a pixel belongs to the foreground (the object) or the background (everything else). Let us first look at a simpler version of this problem, where for those pixels that we do have foreground/background information, we are completely confident in our assignments.

Suppose we have some information about which pixels in an image $I(x, y)$ are part of the foreground and which are part of the background. We can represent this as:

$$A(x,y) \quad = \quad \begin{cases} -1, & I(x,y) \text{ is background} \\ 0, & I(x,y) \text{ is unassigned} \\ 1, & I(x,y) \text{ is foreground} \end{cases}$$

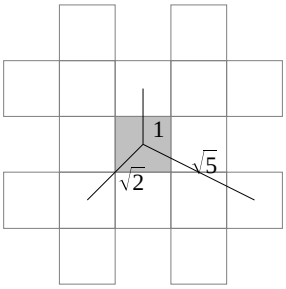


Figure 3.6: The segmentation algorithm is sensitive to the length of the perimeters around foreground regions. It is important that the local pixel connectivity not be so sparse as to introduces artifacts into that perimeter. For example, suppose we just used 4-connected regions. The cost of a zig-zag approximation to a diagonal edge would be $\sqrt{2} = 1.41$ times what it ought to be. 8-connected regions are better, but still distort the perimeter cost significantly, up to a factor of $\frac{1+\sqrt{2}}{\sqrt{5}} = 1.08$. The neighborhood shown here, which is 8-connected plus "knight moves", introduces a distortion of at most $\frac{1+\sqrt{5}}{\sqrt{10}} = 1.02$. Further increases in neighborhood size increases computation time without bringing significant benefit.

We now wish to assign *every* pixel in the image to foreground or background as best we can with the sparse evidence we have. One approach would be to create a cost function to evaluate potential segmentations, and choose the segmentation with minimum cost. If we are willing to accept constraints on the kind of cost function we can use, then there is a family of maximum-flow/minimum-cut algorithms that can provide good approximate solutions to this problem (Boykov and Kolmogorov, 2001). To apply them, we need to translate our problem into the form of a graph, as shown in Figure 3.5. Each pixel maps to a node in the graph, and is connected by edges to the nodes that represent neighboring pixels. There are two special nodes corresponding to the labels we wish to assign to each pixel (foreground or background). The problem the minimum-cut algorithms can solve is how to split this graph into two disjoint parts, with the foreground node in one and the background node in the other, such that the total cost of the edges broken to achieve this split is minimized. So our goal should be to assign costs to edges such that a minimum cut of the graph will correspond to a sensible segmentation.

Let $N(x, y)$ be the node corresponding to pixel $I(x, y)$. Let N_{+1} be the node representing the foreground, and N_{-1} be the node representing the background. If we are completely confident in our classification of pixel $I(x, y)$

into background or foreground, we may encode this knowledge by assigning infinite cost to the edge from $N(x,y)$ to $N_{A(x,y)}$ and zero cost to the edge from $N(x,y)$ to $N_{-A(x,y)}$.

$$\begin{aligned} \mathcal{C}(N(x,y),N_{+1}) &= \begin{cases} \infty, & A(x,y)=1 \\ 0, & \text{otherwise} \end{cases} \\ \mathcal{C}(N(x,y),N_{-1}) &= \begin{cases} \infty, & A(x,y)=-1 \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

This will force the minimum-cut algorithm to assign that pixel to the desired layer. In practice, the visual information will be more ambiguous, and these weights should be correspondingly "softer".

Costs also need to be assigned to edges between pixel nodes. Suppose we expect foreground information to be available most reliably around the edges of the object, as is in fact the case for motion data. Then a reasonable goal would be to use the minimum cut to minimize the total perimeter length of segmented regions, and so merge partial boundary segments into their bounding region. To do this, we could simply assign the actual 2D Euclidean distance between the pixels as the cost. This is not quite sufficient if our edge information is noisy, because it permits almost "zero-area" cuts around individual isolated foreground pixels. We need to place an extra cost on cutting around a foreground pixel so that it becomes preferable to group near-neighbors and start generating regions of non-zero area. For this example, we simply double the cost of cutting edges that are connected to pixels known to be foreground or background.

$$\begin{aligned} \mathcal{C}(N(x_0,y_0),N(x_1,y_1)) &= \begin{cases} D, & A(x_0,y_0)=0, \\ & A(x_1,y_1)=0 \\ 2D, & otherwise \end{cases} \\ \text{where } D &= \sqrt{(x_0-x_1)^2+(y_0-y_1)^2} \end{aligned}$$

Edges are only placed between neighboring pixel nodes, to prevent an explosion in connectivity. A neighborhood is defined as shown in Figure 3.6.

Figure 3.7 shows examples of minimum-cuts in operation. The first image (top left) has two (noisy) lines of known foreground pixels, of length w. The minimum cut must place these pixels inside a foreground region. If the regions are disjoint, the total perimeter will be at least $4w$. If the the lines are instead placed inside the same region, the cost could be as little as $2w+2h$ where h is the distance between the two lines, which is less than w. The figure shows that this is in fact the solution the minimum-cut algorithm finds. The next two examples show what this minimum perimeter criterion will group and what it will leave separate. The fourth example shows that

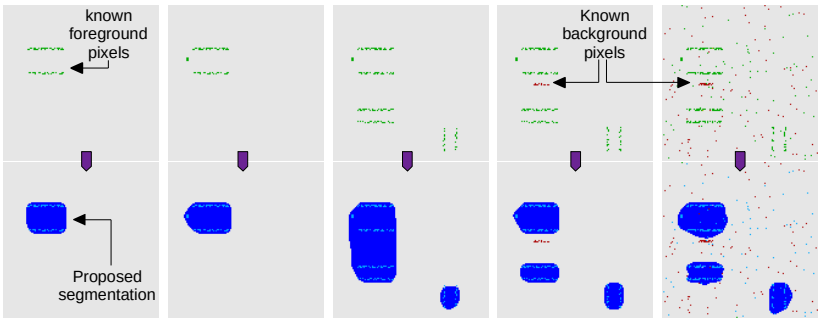
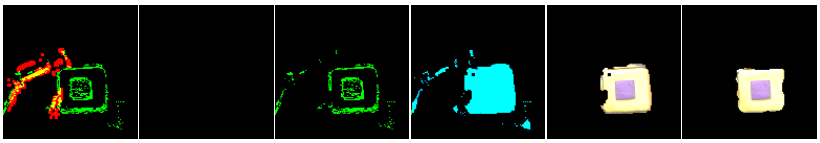


Figure 3.7: Some simple segmentation examples. Input images are shown on the upper row, output is shown as filled regions on the lower row. In the first three cases, the border of the image is set to be background, and the dark pixels are foreground. In the fourth case, a small extra patch of pixels known to be in the background is added, which splits the large segmented region from the previous case in two. The final case shows that the algorithm is robust to noise, where 1% of the pixels are assigned to foreground or background at random. This is in fact a very harsh kind of noise, since we have assumed complete certainty in the data.

by introducing known background pixels, the segmentation can change radically. The patch of background increases the perimeter cost of the previous segmentation by poking a hole in it that is large enough to tip the balance in favor of individual rather than merged regions. This basic formulation can be extended without difficulty to natural data, where foreground/background assignments are soft.

3.6 Before and after

The previous section showed that if there is some evidence available about which pixels are part of the foreground and which are part of the background, it is straightforward to induce a plausible segmentation across the entire image. Figure 3.8 shows an example of how the necessary visual evidence is derived in practice. The statistical significance of changes in pixel values (the "apparent motion") is measured in the frames directly following the contact event, using the continuously updated Gaussian models. The measurements are combined over two frames to avoid situations where the contact event occurs just before the first frame, early enough to generate enough motion for the contact event to be detected but late enough not to generate enough



Motion in frame immediately after impact | *Aligned motion from before impact* | *Masking out prior motion* | *Result of segmentation* | *Largest connected region found* | *Refinement of segmentation*

Figure 3.8: Collecting the motion evidence required for segmentation. The apparent motion after contact, when masked by the motion before contact, identifies seed foreground (object) regions. Such motion will generally contain fragments of the arm and environmental motion that escaped masking. Motion present before contact is used to identify background (non-object) regions. This prevents the region assigned to the object motion from growing to include these fragments. The largest connected region, with a minor post-processing clean-up, is taken as the official segmentation of the object.

motion for a successful segmentation. The frames are aligned by searching for the translation that best matches the apparent motion in the two frames (rotation can be neglected for these very short intervals). A similar measurement of apparent motion from immediately before the contact event is also aligned, and is used to partially mask out motion belonging to the robot arm, its shadow, and unrelated movement in the environment. The remaining motion is passed to the segmentation algorithm by giving pixels a strong "foreground" allegiance (high cost on edge to special foreground node). Importantly, the motion mask from before contact is also passed to the algorithm as a strong "background" allegiance (high cost on edge to background node). This prevents the segmented region from growing to include the arm without requiring the masking procedure to be precise. The maximum-flow implementation used is due to (Boykov and Kolmogorov, 2001).

Perimeter-minimization seems particularly appropriate for the kind of motion data available, since for textureless objects against a textureless background (the worst case for motion segmentation) motion is only evident around the edges of the object, with a magnitude that increases with the angle that edge makes to the direction of motion. A textured, cluttered background could only make life simpler, since it makes it easier to confidently assert that background regions are in fact not moving.

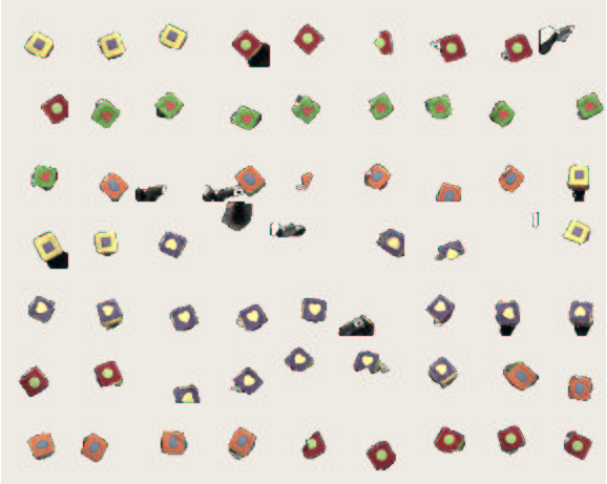


Figure 3.9: Results of a training session, where a toy cube was repeatedly offered to the robot for poking. Each image of the cube corresponds to the segmentation found for it during a single poke. The most common failure mode is inclusion of the robot arm in the segmentation.

3.7 Experimental results

How well does active segmentation work? The segmentation in Figure 3.8 is of the object shown in the introduction (Figure 3.2), a cube with a yellow exterior sitting on a yellow table. Active segmentation has a clear advantage in situations like this where the color and texture difference between object and background would be too small for conventional segmentation but is sufficient to generate apparent motion when the object is poked. Figure 3.11 shows poking from different directions. Figure 3.9 shows about 60 successive pokes of the cube, to give a sense of the kinds of errors that occur. Figures 3.10 and 3.7 shows results for particularly difficult situations. Figure 3.13 shows the area plotted against the second Hu moment (a measure of anisotropy) for a set of four objects that were poked repeatedly. The second Hu moment Φ_2 for a region R with centroid (x_0, y_0) and area μ_{00} is:

$$\begin{aligned} \Phi_2 &= (\nu_{20} - \nu_{02})^2 + 4\nu_{11}^2 \\ \nu_{pq} &= \frac{1}{\mu_{00}^2} \int\int_R (x - x_0)^p (y - y_0)^q dx dy \end{aligned}$$

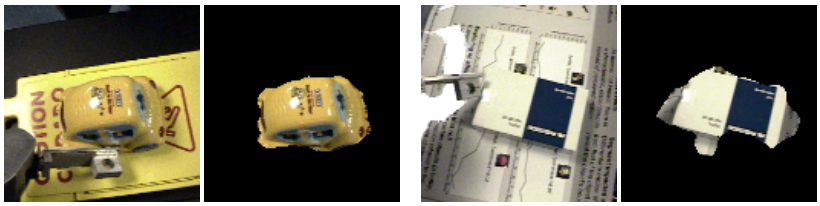


Figure 3.10: Challenging segmentations. The example on the right, a blue and white box on a glossy poster, is particularly difficult since it has complex shadows and reflections, but the algorithm successfully distinguishes both the blue and white part of the box from the background.

If these two features are used to build a simple nearest neighbor classifier, leave-one-out cross validation gives a classification accuracy of 89.8% (chance level is about 25%). So the shape information is a good predictor of object identity.

Qualitatively, poking turned out to be quite a robust procedure, with data gathered opportunistically during the unconstrained interaction of a human with the robot. For example, while the robot was being trained, a teenager visiting the laboratory happened to wander by the robot, and became curious as to what it was doing. He put his baseball cap on Cog's table, and it promptly got poked, was correctly segmented, and became part of the robot's training data.

3.8 Future directions

A unique advantage robots have for object segmentation is that they can reach out and touch the world. Imagine the classical face/vase illusion – this is trivial to resolve if you can simply poke it to see which part is free space and which is not. But poking is simply the lowest-hanging fruit in the set of active strategies a robot could use for achieving object segmentation. If the robot is unsure where the boundaries of an object lie, here are some strategies it can use :-

1. Poke the object gently. Tapping a solid object will induce a small motion of that object. This will result in a coherent region of optic flow on the image plane. If the object is non-rigid, or attached to other objects, then the response will be messy and complicated – but this is in some sense inevitable, since it is in just such cases that the idea of a unique "object boundary" runs into trouble

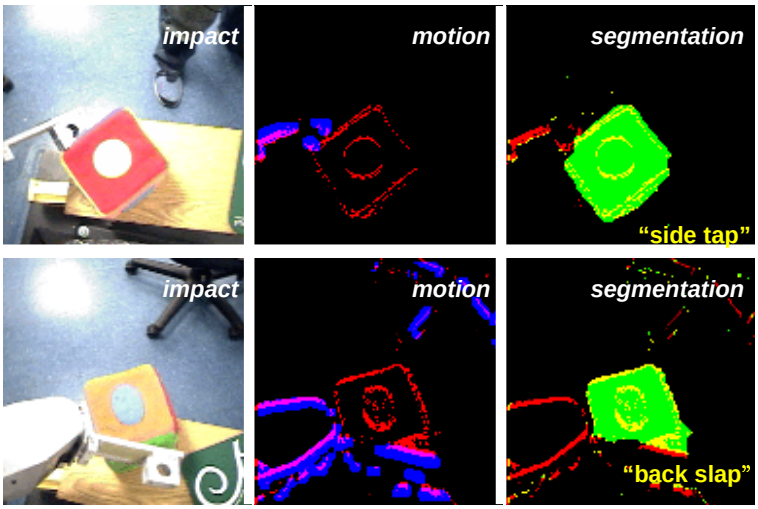


Figure 3.11: Cog batting a cube around from different directions. The images in the first column are from the moment of collision between the arm and the cube, which is detected automatically. The middle column shows the motion information at the point of contact. Red is new motion, purple and blue are pre-existing motion. Notice in the bottom row that the chair is moving. The bright regions in the images in the final column show the segmentations produced for the object.

2. Thump the object savagely. A big disturbance is apt to generate a confusing motion that is hard to process directly. But it will move the object away from its local surroundings, giving another "role of the dice" – an opportunity for the robot to see the object against a new background, perhaps with better contrast. Frequently visual ambiguity is only a local, accidental effect.

3. Try to get the arm's endpoint beside the object. Anywhere the endpoint can reach is presumably free space, constraining the boundary of the object. We can use the arm's endpoint as a mobile reference object to confirm our theories of where free space lies.

4. Try to get the arm's endpoint behind the object. This has the advantage of putting a known background behind the object. Imagine the arm painted bright red to see the advantage of this for identifying the object's boundary.

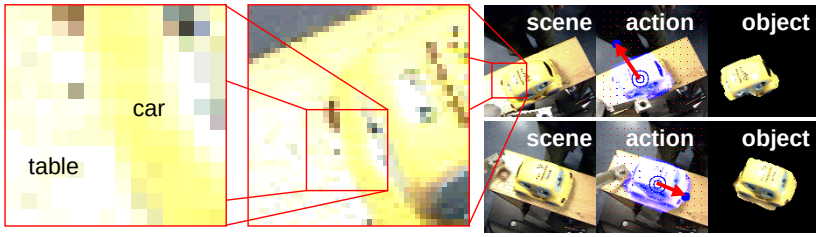


Figure 3.12: An example of the power of active segmentation. The images marked "scene" show two presentations of a yellow toy car sitting on a yellow table. The robot extends its arm across the table. In the upper sequence it strikes from below, in the lower sequence it strikes from the side ("action" images). Once the arm comes in contact with the car, it begins to move, and it can be segmented from the stationary background ("object"). On the left of the figure, a zoomed view of the car/table boundary is shown – the difference between the two is very subtle.

5. Ask the human to present the object. A human bringing an object near the robot offers the dual advantage of motion cues and a known (if complicated) partial background – the hand.

6. Another alternative is to displace the robot's own head and body, again to get another "role of the dice", or to access three-dimensional information over a longer baseline than is available from the stereo cameras.

This does not even begin to exhaust the space of active strategies that are possible for object segmentation, and at the Humanoid Robotics Group at MIT we are investigating several others (Arsenio et al., 2003).

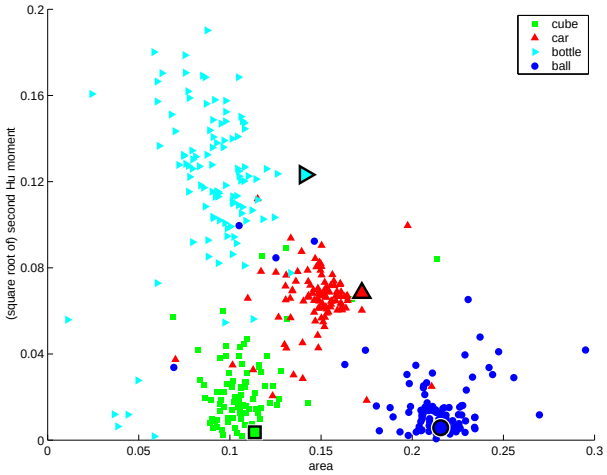


Figure 3.13: A large collection of segmentations are grouped by object identity, and then plotted (area versus second Hu moment). The enlarged markers show hand-segmented reference values. The segmentations are quite consistent, although area tends to be a fraction smaller than in the hand-segmented instances.

CHAPTER 4

The outer limits: learning about edges and orientation

> *The Disc, being flat, has no real horizon. Any adventurous sailors who got funny ideas from staring at eggs and oranges for too long and set out for the antipodes soon learned that the reason why distant ships sometimes looked as though they were disappearing over the edge of the world was that they* were *disappearing over the edge of the world.* (Pratchett, 1986)

The previous chapter showed how elementary sensitivity to motion is sufficient to gather segmentations of objects in the robot's vicinity, with some support from the robot's behavior to evoke easily processed scenarios. Once this data is coming in, there is a lot that can be learned from it. One reasonable use of the data would be to learn about the appearance of specific objects, and the next chapter (Chapter 5) will address that. But even before that, it is also possible to simply learn something about the appearance of boundaries, since the robot now has a collection of such boundaries side by side with their visual appearance. In particular, this allows an orientation detector to be trained on automatically annotated data. Orientation information is present in images at all scales. It is typically detected using quadrature filters applied at many locations and scales (Freeman, 1992), an approach developed to be independent of contrast polarity and to act equally well on edges and

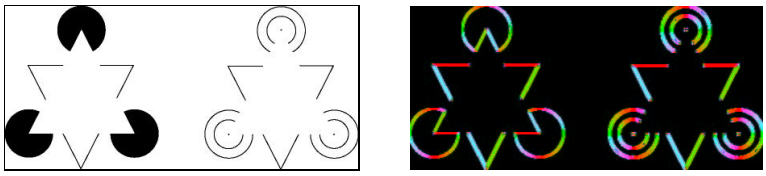


Figure 4.1: The goal of orientation detection is to take an image such as the one shown on the left here, and annotate every point in it with a direction, if there is a well-defined orientation that can be associated with it. For example, on the right is a color-coded (if viewed in color) orientation map corresponding to the first image, where all horizontal lines and edges are colored red, etc. This map is produced by the methods developed in this chapter. It shows only orientation that is clear from local information – the "illusory contours" present in the Kanizsa triangles are not detected.

lines. With the data the robot collects, the opportunity arises to take a complementary, empirical approach, where the appearance of edges is learned from experience rather than derived theoretically. The main challenge is whether the appearance of edges can be sampled densely enough to get good coverage on a reasonable timescale. The answer to this is shown to be yes, primarily because orientation information is quite robust to pixel-level transformations. It turns out that a useful orientation filter can be constructed a a simple interpolating look-up table, mapping from a very small window size (4×4 pixels) directly to orientation. This allows for extremely rapid access to orientation information right down at the finest scale visible.

The contribution of this work is to demonstrate that orientation detection is amenable to empirical treatment, and that it can be performed at a very high speed. This work is critical to a real-time implementation of the object recognition method that will be proposed in Chapter 5.

4.1 What is orientation?

Natural images are full of discontinuities and local changes. This anisotropy can be used to associate directions with regions of the image. These directions are potentially more robust to image-wide transformations than the individual pixels upon which they are based. The most obvious example is a luminance edge, where there is a discontinuity between a dark and light region. The direction associated with this edge remains unchanged even if overall illumination on the regions change their appearance dramatically. Contours of constant luminance on a shaded surface behave somewhat like edges also,

with luminance change being minimal parallel to the contour and maximal when measured perpendicular to them. For such directional changes in luminance, or any other property, it is natural to associate a direction or *orientation* in which change is minimal. In this chapter, we will be concerned with the orientation associated with edges in luminance at the finest scale available. This is certainly not all that is to be said about orientation (see, for example, Figure 4.1). But it is a useful case, particularly for object localization and recognition. Orientation detection will prove key to achieving orientation and scale invariance in these tasks.

Orientation is associated with neighborhoods rather than individual points in an image, and so is inherently scale dependent. At very fine scales, relatively few pixels are available from which to judge orientation. Lines and edges at such scales are extremely pixelated and rough. Orientation filters derived from analytic considerations, with parameters chosen assuming smooth, ideal straight lines or edges (for example, Chen et al. (2000)) are more suited to larger neighborhoods with more redundant information. For fine scales, an empirical approach seems more promising, particularly given that when the number of pixels involved is low, it is practical to sample the space of all possible appearances of these pixels quite densely. At very fine scales, the interpretation of an image patch could hinge on a relatively small number of pixels. Noise sensitivity becomes a critical issue. But even beyond that, it seems that the assignment of labels to image patches is likely to be quite a non-linear process.

4.2 Approaches to orientation detection

Most methods for detecting local orientation fall into one of two categories. Gradient-based approaches such as that of Kass and Witkin (1987) are relatively direct, and operate by applying spatial derivatives to the output of an isotropic edge-detecting filter (such as a Laplacian or difference of Gaussians). A different approach often used is to examine the response of each neighborhood in the image to a set of oriented filters, chosen so that some of them respond to edges ('cosine-phase'), and some respond to bars ('sine-phase'), analogous to the receptive fields found by Hubel and Wiesel in the visual cortex of cats (Hubel and Wiesel, 1962). The filter set may be overcomplete and non-orthogonal since image reconstruction is not the goal. Figure 4.2 shows an example of a possible filter set. If the filter is chosen carefully, then it need only be replicated at a discrete number of orientations, and the response of the image to any other orientation computed from the response to those few. Such filters are said to be steerable (Freeman and Adelson, 1991). Orientation is computed by finding the orientation that maximizes the

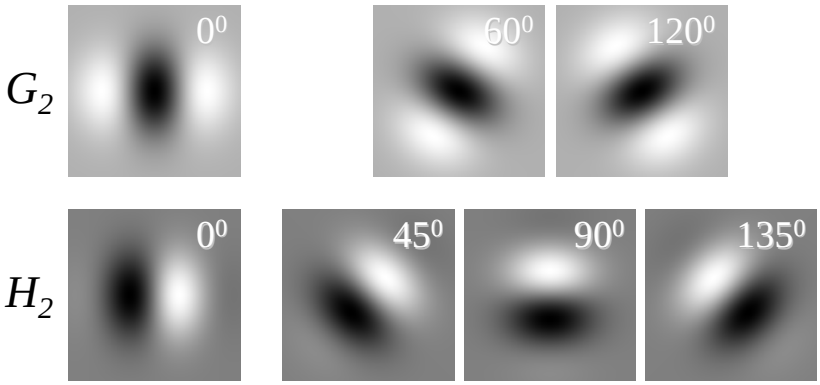


Figure 4.2: An example of a steerable filter, following Freeman and Adelson (1991). G_2 is the second derivative of a Gaussian, and H_2 is an approximation to its Hilbert transform. These two filters are said to be in quadrature. From its even form, G_2 will respond well to vertical lines. H_2 is odd, and will respond well to vertically oriented step edges. The theory associated with steerable filters shows that the response of an image with a small set of basis filters at discrete angles, as shown here, can be used to compute the response of one of the filter rotated to any angle. Orientation detection then becomes applying these filters and computing the angles that would give maximum response.

response of the image to the filter (here the cosine-phase and sine-phase filters can be thought of as the real and imaginary components of a single quadrature filter).

4.3 Empirical orientation detection

Poking allows the robot to build up a reference "catalog" of the manifold appearances real edges can take on. At fine scales, with relatively few pixels, we can hope to explore the space of possible appearances of such a neighborhood quite exhaustively, and collect empirical data on how appearance relates to orientation. This chapter is basically an exploration of how edges in "natural" images appear when viewed through an extremely small window (4 by 4 pixels). This window size is chosen to be large enough to actually allow orientation to be well-defined, but small enough for the complete range of possible appearances to be easily characterized and visualized. Even at this scale, manual data collection and labelling would be extremely tedious, so it is very advantageous to have a robot to take care of this. The robot automat-

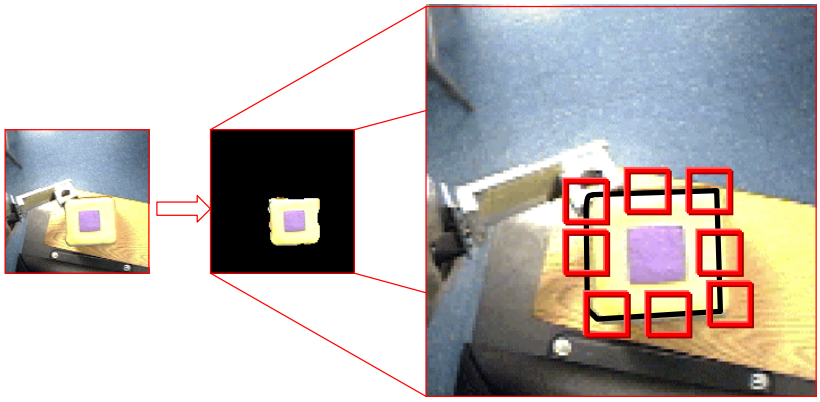


Figure 4.3: Sampling the appearance of edges at an object boundary. The object is detected and segmented as described in Chapter 3. Its boundary is sampled, and quantized window appearance is stored along with the actual angle of the boundary at that point.

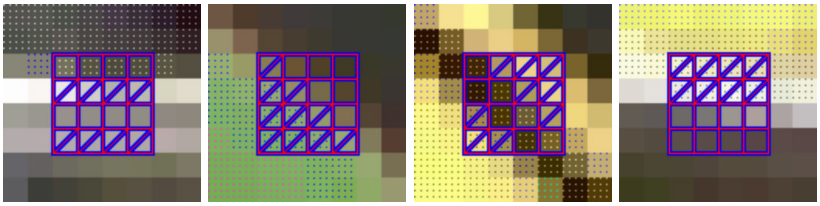


Figure 4.4: Some examples of boundary samples. Dotted pixels belong to a segmented object. The four-by-four grid overlaid on the boundary shows the result of thresholding.

ically compiles a database of the appearance of oriented features using the poking behavior.

Oriented features were extracted by sampling image patches along object boundaries, which were in turn determined using active segmentation. The resulting "catalog" of edge appearances proved remarkably diverse, although the most frequent appearances were indeed the "ideal" straight, noise-free edge (Section 4.3). Finally, it is a simple matter to take this catalog of appearances and use it as a fast memory-based image processing filter (Section 4.3).

The details of the robot's behavior are as described in Chapter 3, and are briefly reviewed here. A robot equipped with an arm and an active vision head was given a simple "poking" behavior, whereby it selected objects in

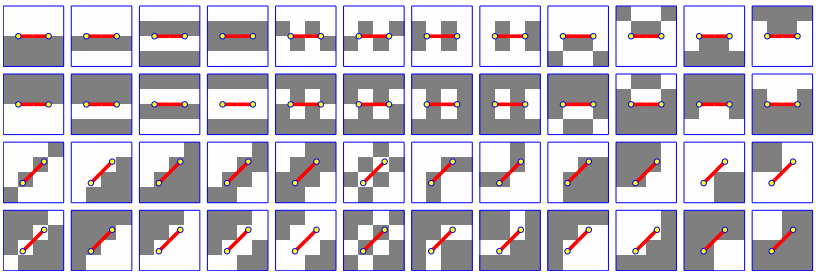


Figure 4.5: Edges have diverse appearances. This figure shows the orientations assigned to a test suite prepared by hand. Each 4×4 grid is a single test edge patch, and the dark line centered in the grid is the orientation that patch was observed to have in the training data. The oriented features represented include edges, thin lines, thick lines, zig-zags, corners etc.

its environment, and tapped them lightly while fixating them. As described in Chapter 3, the motion signature generated by the impact of the arm with a rigid object greatly simplifies segmenting that object from its background, and obtaining a reasonable estimate of its boundary. Once this boundary is known, the appearance of the visual edge between the object and the background can be sampled along it (see Figure 4.3). These samples are labelled with the orientation of the boundary in their neighborhood (estimated using a simple discrete derivative of position along the boundary). The samples are assumed to contain two components that are distinguished by their luminance. The pixels of each sample are quantized into binary values corresponding to above average and below average luminance. Quantization is necessary to keep the space of possible appearances from exploding in size. The binary quantization gives a very manageable 65536 possible appearances. About 500 object boundaries were recorded and sampled. 49616 of the possible appearances (76%) were in fact observed; the remaining 24% were all within a Hamming distance of one of an observed appearance. The orientation of these unobserved appearances were interpolated from their immediate neighbors in Hamming space. If the same appearance was observed multiple times, the orientations associated with these observations are averaged using a double-angle representation (Granlund, 1978).

It is a straightforward matter to use the data we have collected to filter an image for fine scale orientation features. A 4×4 window is moved across the image, sampling it as described earlier in Section 4.3. Each sample is used as an index into a table mapping appearance to orientation.

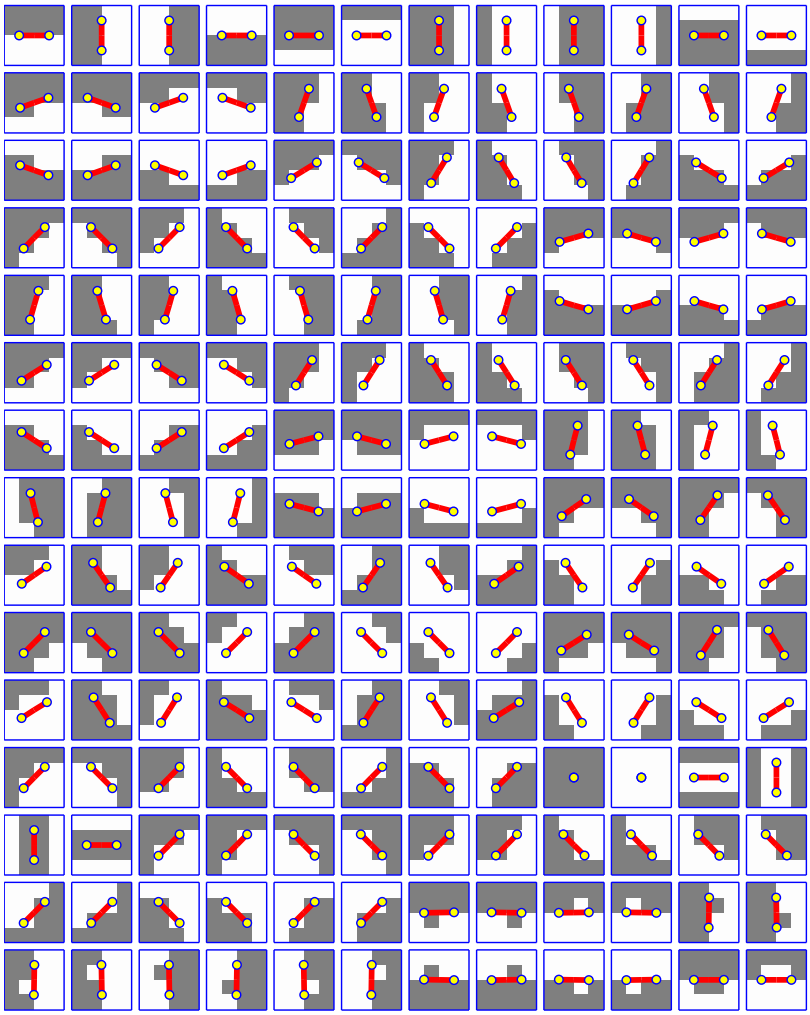


Figure 4.6: The most frequently observed edge appearances. All patches observed are replicated for all 90° rotations, mirror flips, and inversion of foreground/background. The most frequent (top) are simple straight edges. The line in the center of each patch shows the orientation associated with that patch. After the straight edges, the completely empty patch is common (produced in saturated regions), followed by a tube-like feature (third-last row) where the boundary is visually distinct to either side of the edge. This is followed corner-like features and many thousands of variations on the themes already seen.

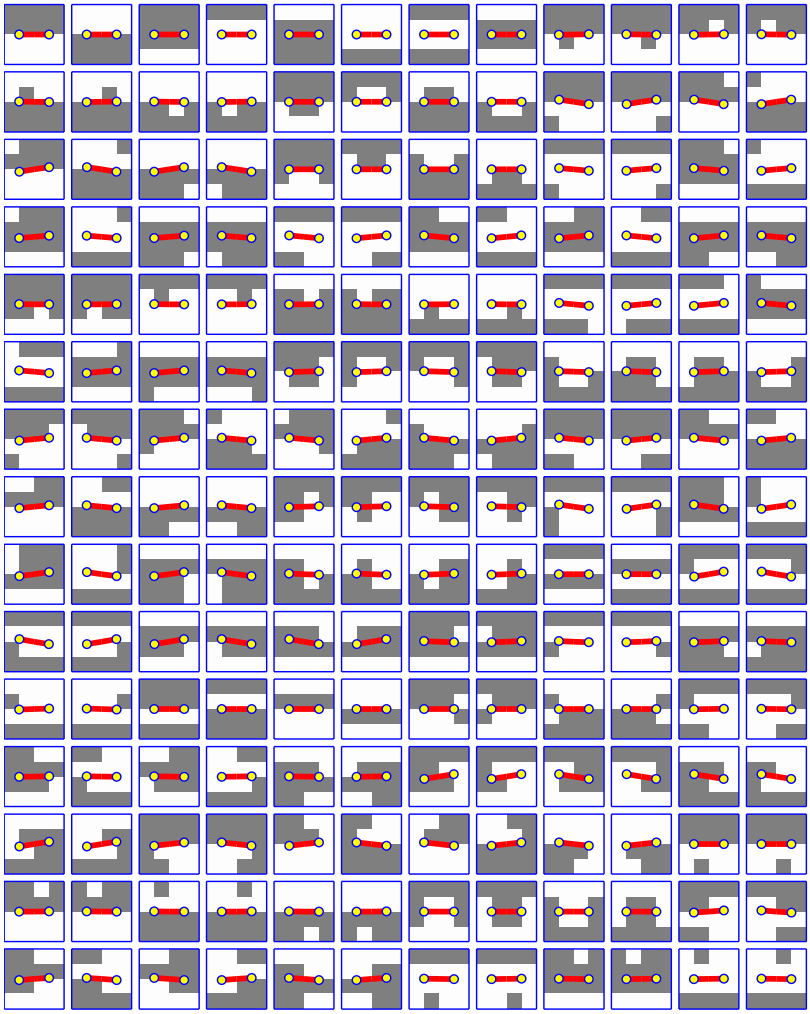


Figure 4.7: The most frequently observed appearances whose orientation is within 5° of the horizontal. There is a clear orientation assigned to many patches that deviate a great deal from "ideal" edges/lines, showing a robustness that is examined systematically in Figure 4.5.

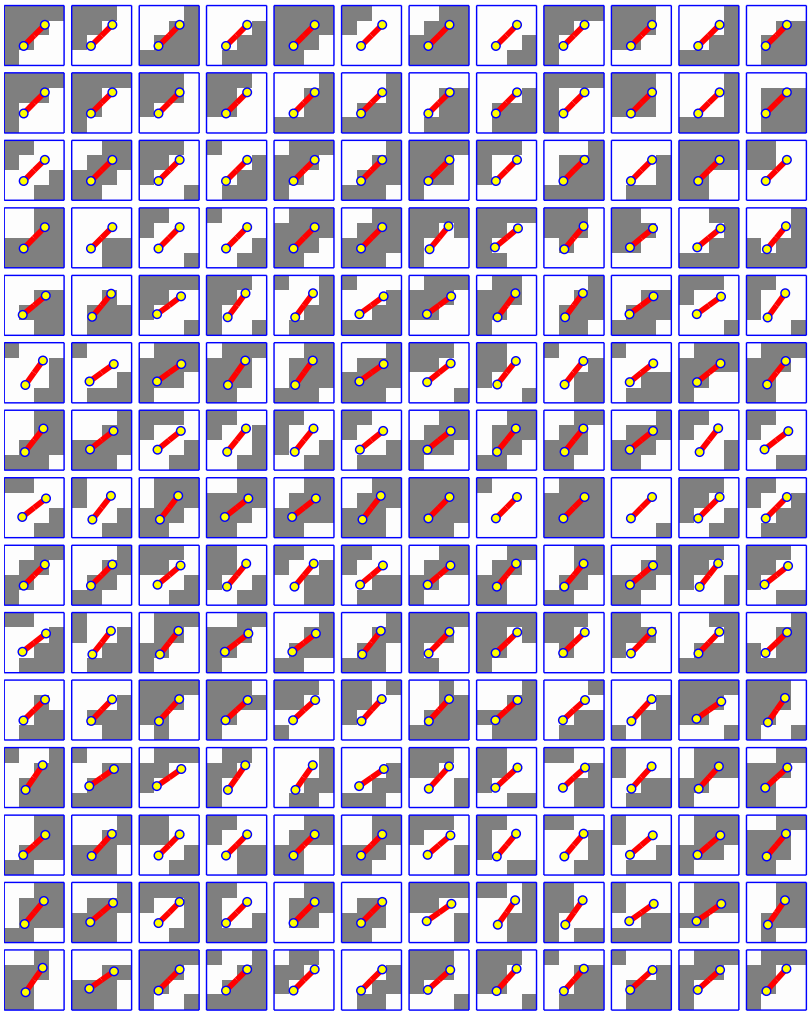


Figure 4.8: The most frequently observed appearances whose orientation is in the range $40 - 50°$. Again, there is a clear orientation assigned to many patches that deviate a great deal from "ideal" edges/lines.

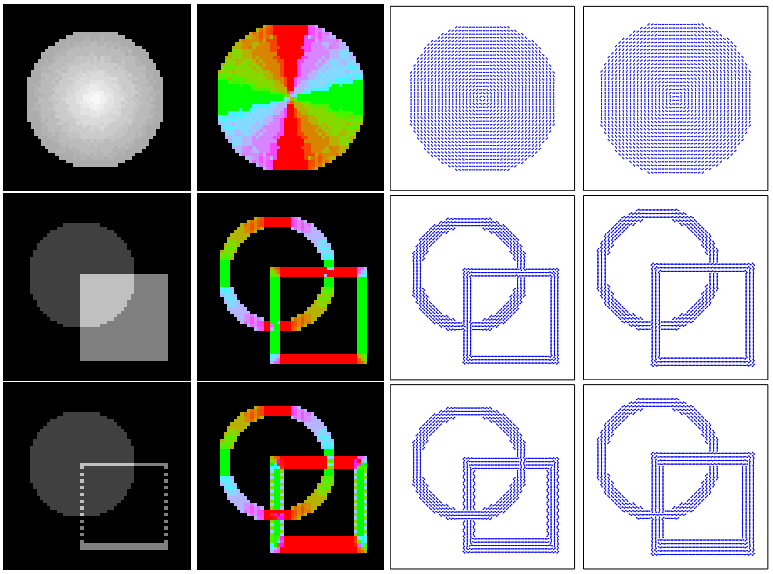


Figure 4.9: The orientation filter applied to some synthetic test images (on left), modeled after an example in (Freeman and Adelson, 1991). The second column shows the output of the orientation filter, color coded by angle (if viewed in color). The third column shows the same information in vector form. The fourth column shows the orientation determined using steerable quadrature filters Folsom and Pinter (1998) applied on the same scale. The results are remarkably similar, but the quadrature filters are much more computationally expensive to apply.

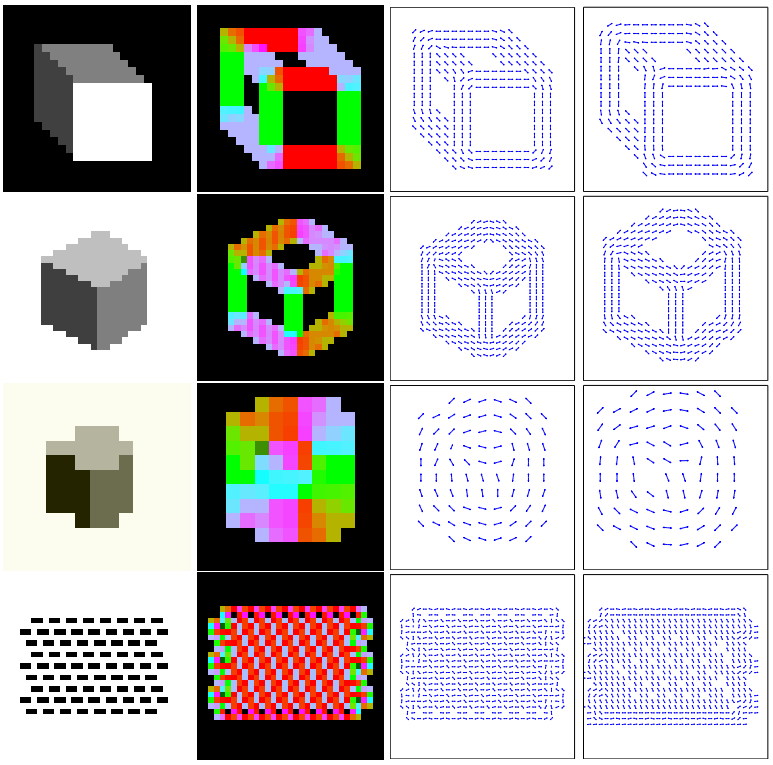


Figure 4.10: Some more test images, but on a much smaller scale – the individual pixels are plainly visible, and no smoothing is applied. These tests are modeled after an example in (Folsom and Pinter, 1998), but significantly scaled down.

4.4 Results

Figure 4.5 shows that although the data collection procedure operates on views of simple physical edges, the appearance of these edges can be quite complex. Nevertheless, the most common appearances observed are ideal, noise-free edges, as Figure 4.6 shows. The first four appearances shown (top row, left) make up 7.6% of all observed appearances by themselves. Line-like edges are less common, but do occur, which means that it is perfectly possible for the surfaces on either side of an edge to be more like each other than they are like the edge itself. This was completely serendipitous – it was anticipated that obtaining and automatically labelling such examples would be very difficult.

Figure 4.5 shows the most frequently occurring image appearances with a particular orientation. Here it is clearer that the most frequent patches are generally "ideal" forms of the edges, followed by very many variations on those themes with distracting noise. Amidst the edge-like patterns are examples of a line with single-pixel thickness, and a pair of such lines running parallel. It is encouraging that examples of such appearances can be collected without difficulty and united with more classical edge patches of the same orientation.

Figure 4.9 shows the orientations measured for a 64×64 image consisting of a circle and square. This is based on an example in (Freeman and Adelson, 1991). The detector gives good results for solid edges with arbitrary contrast, and various kinds of lines. The response to edges is diffuse by design – during data collection, samples are taken both along the boundary and slightly to either side of it, and treated identically. If a sharper response is desired, these side-samples could be dropped, or their offset from the boundary could be recorded. Figure 4.10 shows the filter operating on a very small image of a cube. Each visible edge of the cube is clearly and faithfully represented in the output.

Figure 4.13 systematically explores the effect of adding noise to an "ideal" edge. The resilience of the orientation measure is encouraging, although a small number of gaps in coverage are revealed, suggesting that further data should be collected.

4.5 Discussion and Conclusions

The orientation detection scheme presented in this chapter has an unusual combination of properties, some of which are essential to the approach and some which are incidental details :-

- ▷ **Data driven (versus model based).** Detection relies heavily on the existence of training data – it is not achieved directly based on a formal

model of edges instantiated in an algorithm.

- ▷ **Uses look-up table (versus neural network, support vector machine, etc.).** The use of training data is simply to populate a look-up table, rather than anything more elaborate.
- ▷ **Autonomous data collection (versus human annotation).** Training data is collected by the robot, and not a human.

Data driven

Work on edge and orientation detection has historically been model based, rather than data driven. To make progress analytically, the nature of edges was grossly simplified – for example, researchers worked with additive Gaussian noise overlaid on a luminance step (see, for example Canny (1986)). Before long it was pointed out that edges can take a diversity of forms beyond steps or lines (Perona and Malik, 1990). With the introduction of diverse cases, an empirical approach becomes more attractive. Jitendra Malik's group are now looking at how to locate boundaries between objects in images using features trained on human-produced segmentations (Martin et al., 2002). Many other parameters of a modern edge detector can also profit from empirical training, and can be optimized per domain (Konishi et al., 2003). So there is clearly considerable scope for a data driven approach to edge detection to improve performance.

Look-up table

The use of look-up tables has an important place in AI and computer science, from the Huge Look-Up Table problem in philosophy (Block, 1978) to the implementation of elementary arithmetic operations in CPU design (Wong and Flynn, 1992). An interpolating look-up table is also just about the simplest possible learning module. When it can be used, the results of learning are much simpler to understand than is the case for neural networks, support vector machines, etc. For example, the work by McDermott (2000) on training a neural network to detect junctions ran into the problem that, even with sensitivity analysis, it can be hard to understand a network's failure modes. With a look-up table, it is trivial. Chapter 4 exploited this fact to provide several visualizations of cross-sections of the look-up table. The index into the look-up table used is quantized pixel values. This stays closer to the raw image than other work (Konishi et al., 2003; Martin et al., 2002) which focuses on optimizing the combination of existing hand-designed features. In theory, this means the approach could capture unanticipated domain-specific properties that will not show up in the hand-designed cases. This possibility was not explored here since the work was implemented on a robot inhabiting

a single fixed space.

Populating a look-up table makes for fast run-time operation. It is possible to make a filter bank approach that runs at comparable speeds – for example, orientation can be detected from the output of two 3×3 filters, if we are willing to put some work into interpreting the different responses to step and line edges (essentially line edges give doubled responses, and appear to be a pair of close step edges). The look-up table approach encapsulates this interpretation step automatically, since it is trained on the end-to-end judgement required (from pixels to angles).

Autonomy

Training examples of edges could be generated in many ways. For example, computer graphics could be used to make images with known ground truth, or human labelled datasets could be used as in Martin et al. (2002) and Konishi et al. (2003). The work of Konishi et al. (2003) has shown that domain-dependent improvements can be made in edge detection, so in that sense an argument can be made for adaptivity. An autonomous, empirical approach holds out the promise of developing a 'wise' low-level perceptual system that makes good context-sensitive guesses, making the job of higher level modules that much simpler. This was beyond the scope this thesis, but it was certainly a motivating consideration and a direction for future work.

While this work has shown that idealized edges are well grounded empirically, in that they occur more frequently than other variants, it also shows that many more exotic forms do occur and can profitably be modeled. At fine scales, where the number of pixels used to compute orientation is low, a practical approach is to sample the appearance of edges empirically and average over noise (see Figure 4.17). With the large cache size of modern processors, this memory-based approach to orientation detection can facilitate extremely rapid orientation detection, which is important for real-time vision systems (Kubota and Alford, 1995).

Rectangular windows are the natural size for real-time machine vision applications. The memory-based approach proposed here has the advantage that it can make use of every pixel in the window a principled way. Filters for orientation detection are typically circular in nature, and so must ignore pixels that lie outside the largest circle that fits inside the window.

Can this technique be applied to larger windows? Not easily. In fact the window size used in one of the earliest papers on orientation detection, which had a diameter of eight pixels, seems completely out of reach (Hueckel, 1971). A 5×5 window of binary pixels can take on $2^{5\times5}$ possible values – about 33.6 million, or about 2.1 million allowing for symmetries. This would be hard to sample exhaustively, but with some further quantization a look-up

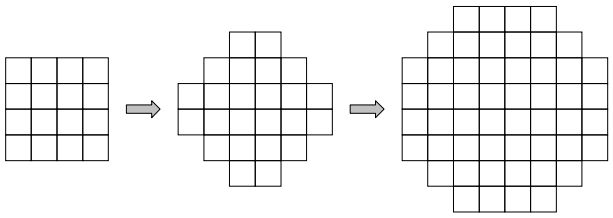


Figure 4.11: Possibilities for expanding the size of the orientation filter.

table of that size would not be impossible. An intermediate possibility shown in Figure 4.11 involves one fewer pixel, and has a more symmetric shape: a 4×4 window augmented with 8 extra pixels to "round it off".

Would an empirical approach work for features other than orientation? This isn't clear, since not all features are as robust to pixel-wise transformation as orientation is – and hence it may not be as easy to explore their space of appearances as exhaustively.

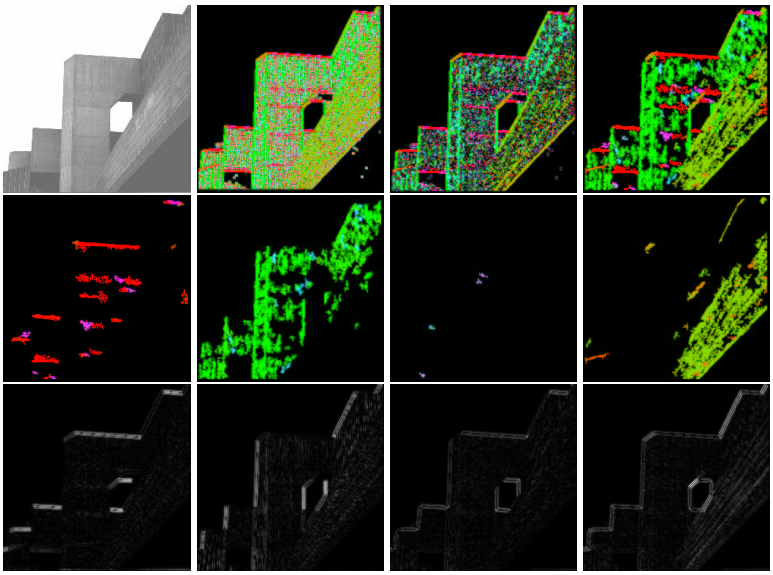


Figure 4.12: Here the orientation detector is applied to an image from Chabat et al. (1999). The top row shows from left to right the original image, output of the Folsom detector, output of the memory-based detector, and a simple enhancement of that output using region growing. The second row filters that output by orientation range (within 22.5° of horizontal, vertical, $+45^\circ$ and -45° respectively). The final row shows the output of a steerable filter by comparison, steered to the same nominal orientations, to reinforce that orientation does not pop out immediately from those filters – note for example the significant response in the third column at -45° even though there is nothing at that orientation in the image; they require a level of additional processing that the memory-based approach bypasses.

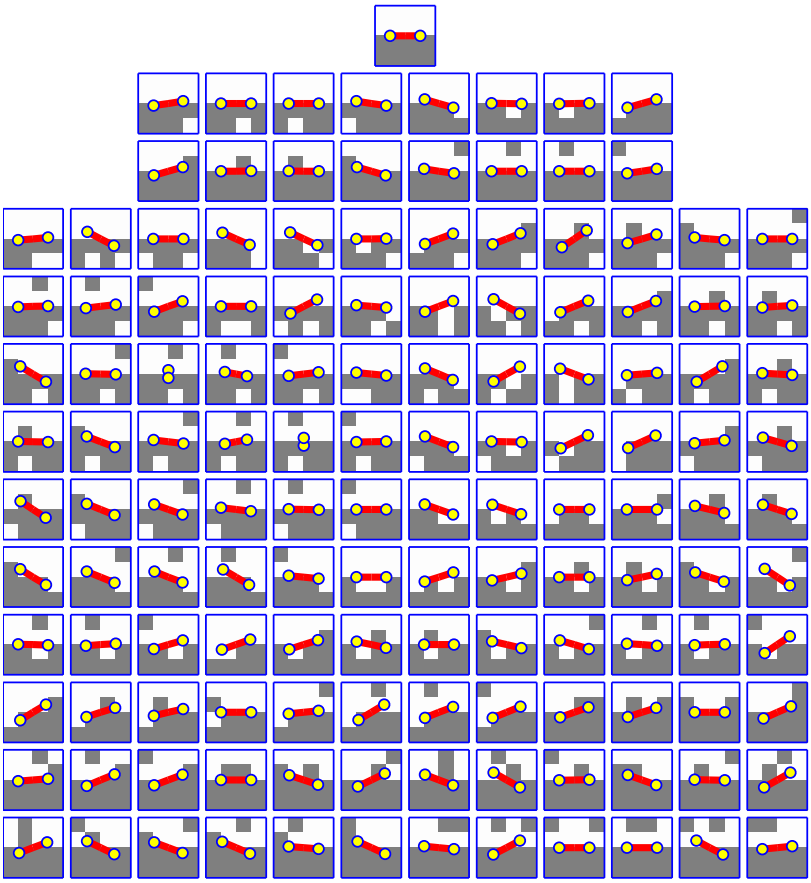


Figure 4.13: The top row shows an “ideal” view of a horizontal edge. The next two rows show patches with a single pixel perturbation from this ideal. The estimated angle remains close to horizontal. The rows that follow show 2-pixel perturbations (25% of the pixel weight of the edge). The behavior of the orientation estimate is generally reasonable. In two cases there is insufficient data for a good estimate (just one sample is available).

Figure 4.14: Perturbations from an “ideal” view of a diagonal edge. There are now several cases in which little training data is available, and one example of a divergent direction estimate (fourth row from bottom, third column from right).

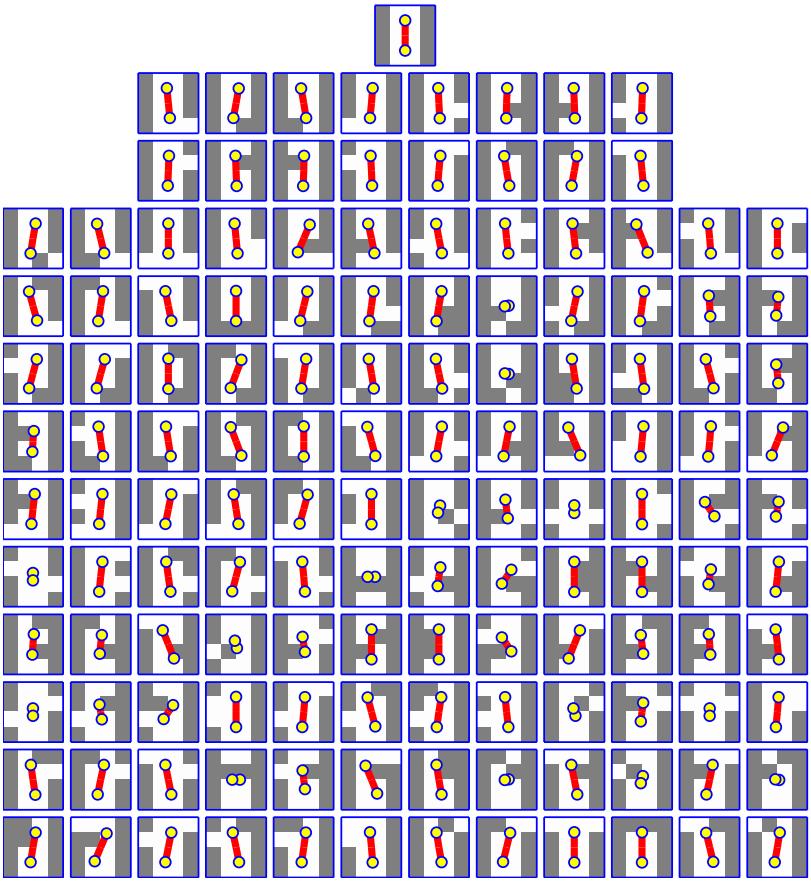


Figure 4.15: Perturbations from an “ideal” view of a tube.

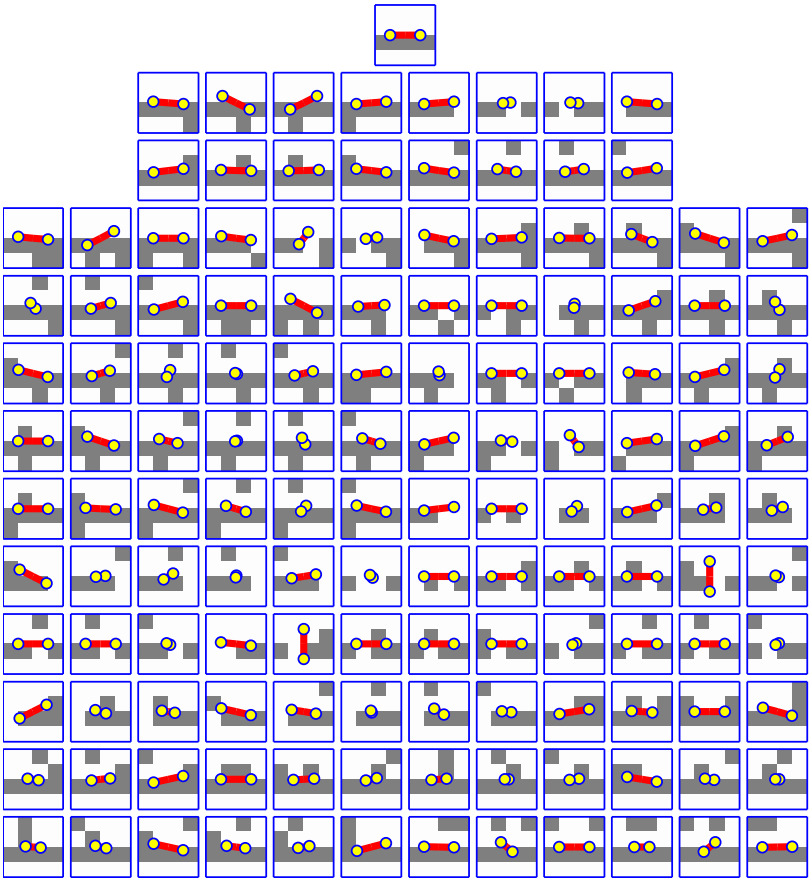


Figure 4.16: Perturbations from an "idea" view of a line. Here the pixel weight of the line is low, so the perturbations have a correspondingly more drastic effect. Lines are also seen less often in the training data, since they require special conditions at the object boundaries sampled (the boundary must be unlike the surface on either side of it). There is considerable room for improvement here if other sources of ground truth could be acquired. For example, orientation information could be propagated across time or space from neighboring patches of known orientation to less frequently encountered patches.

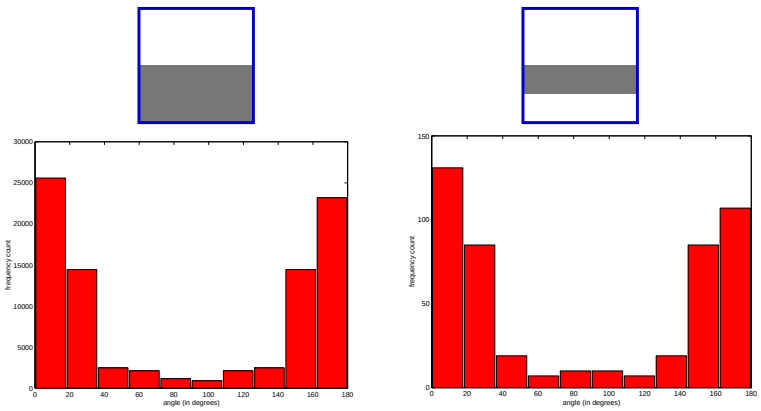


Figure 4.17: The plot on the left shows the frequency with which a thick step edge (as shown in Figure 4.13) is labelled with each possible angle. The distribution has a peak at 0°/180° as is appropriate, but other values do occur. About 7% of the samples lie at close to right angles to the nominally correct value. The plot on the right shows the same results for a thin line (as shown in Figure 4.16). The basic shape is the same, but the pattern occurs much less frequently overall – hundreds of times versus tens of thousands.

CHAPTER 5

Close encounters: recognizing nearby objects without contact

> *The followers of Om had lit their campfire in the crumbled halls of Gilash, just as the Prophet had said, and that counted, even though he'd only said it 5 minutes earlier, when they were looking for the firewood.* (Pratchett, 1992b)

With the active segmentation behavior introduced in Chapter 3, the robot can familiarize itself with the appearance of nearby objects by poking them. This chapter is concerned with learning to locate, recognize, and segment those objects whenever they are present without further contact.

5.1 Approaches to object recognition

Physical objects vary greatly in shape and composition. This variety is reflected in their visual appearance. Unfortunately, it is not a straightforward matter to recover object properties from appearance, since there are many other factors at work – illumination, relative pose, distance, occlusions, and so on. The central challenge of object recognition is to be sensitive to the identity of objects while maintaining invariance in the face of other incidental properties. There are at least two broad approaches to recognition, geometry-based and appearance-based.

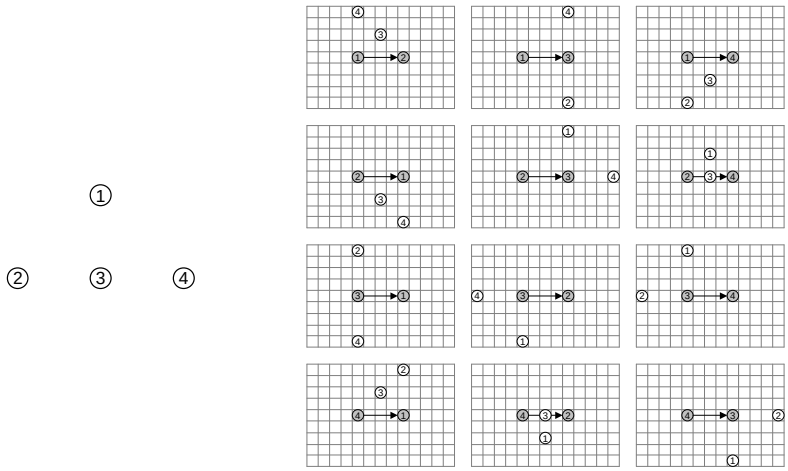


Figure 5.1: Geometric hashing for 2D-2D recognition. For the set of points shown on the left, each pair is considered in turn and used to normalize the rest for translation, orientation, and scale. The normalized locations of the points for each permutation are stored in a hash table, along with references to the model and pair of points that generated them.

5.1.1 Geometry-based recognition

Image formation is a geometric process, so one way to approach recognition is to model invariant geometric relationships that hold for a particular class of object. These relationships can be between points, lines, surfaces or volumes. They may be known for many possible views of the object, or just one. When a new scene is presented, geometric relationships in it are measured and matched against the model. There are many details in what to measure and how to do the matching (there is a good review in Selinger (2001)). The main difficulty is the combinatorics of the search involved. There are a lot of free parameters to search over when we try to match an unsegmented scene to an object model – in particular, which elements in the scene correspond to the object, and what the transformation is between those elements and the object. For high-speed performance, geometric hashing is a useful technique (for a review see Wolfson and Rigoutsos (1997)). In this method, geometric invariants (or quasi-invariants) are computed from points in model (training) images, then stored in hash tables. Recognition then simply involves accessing and counting the contents of hash buckets. One possibility for the geometric invariants is to take a set of points selected by an interest operator, and use each pair of points in turn to normalize the remainder by scale and

rotation. The position of the normalized points can be stored in a 2D hash table, as shown in Figure 5.1. Invariants in 3D are more difficult to achieve, but various solutions have been proposed.

5.1.2 Appearance-based recognition

While the world is indeed geometric in nature, geometric features are not particularly easy to extract reliably from images. In appearance-based recognition, the focus is shifted from the intrinsic nature of an object to properties that can be measured in images of that object, including geometric properties but also surface properties such as color or texture. For example, Swain and Ballard (1991) proposed using the set of colors present in segmented views of an object as their representation. Regions of an image that contain the same color mix (as determined by histogram intersection) could contain the object. Histogram back-projection can be used to quickly filter out regions unlikely to contain the object, by assigning each pixel a weight based on the frequency of its color in the histogram. Some form of region-growing method then accumulates this evidence to find plausibly colored regions. This method is very fast, and for objects with distinctive colors it can be useful as a pre-filter to more computationally expensive processing. Color is also intrinsically somewhat robust to scale and rotation, but is sensitive to changes in the spectral profile of illumination.

Many appearance-based methods are window-based. A classifier is built that operates on a rectangular region of interest within an image. That window is moved across the entire image, at multiple scales, and sometimes multiple orientations. Responses of the classifier across locations and scales are combined using various heuristics. Variation in orientation (rotation in depth) is typically dealt with by training up multiple recognizers for various poses. These poses can be sampled quite sparsely, but still each pose requires iteration of the search procedure. There are ways to speed all this up, for example using a cascade of classifiers that reject clearly non-target windows early, devoting full analysis only to plausible targets. The actual classifier itself can be based on eigenspace methods or many other possibilities.

5.2 Hashing with rich features

The approach used here is like geometric hashing, but uses richer features that include non-geometric information. Geometric hashing works because pairs of points are much more informative than single points. An ordered pair of points defines a relative scale, translation, and orientation (in 2D). Further points may be needed for more general transformations, such as affine

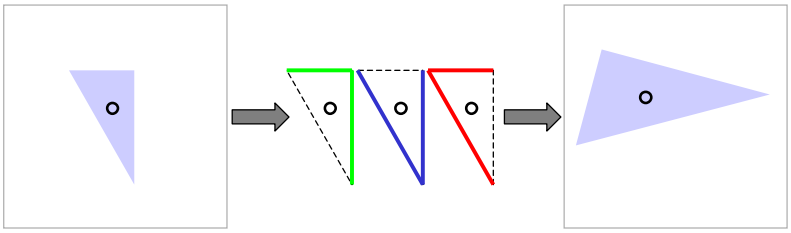


Figure 5.2: Rich features for hashing. Every pair of edges in the object to be recognized are stored, in terms of their relative position and orientation, along with samples of the colors between them. This representation is invariant to translation, scale, and in-plane rotation.

or projective, or to move to 3D (Wolfson and Rigoutsos, 1997). But, even staying with the 2D case, using just two points is somewhat problematic. First of all, they are not at all distinctive – any two points in an image could match any two points in the model. Hashing doesn't require distinctiveness, but it would be a useful pre-filter. Secondly, there is no redundancy; any noise in the points will be directly reflected in the transformation they imply.

One possibility would be to use triplets of points, or more. Then we have distinctiveness and some redundancy. But we also have an explosion in the number of possible combinations. Pairs of points are just about manageable, and even then it is better if they are drawn from a constrained subset of the image. For example, the work of Roy and Pentland (2002) uses histograms of the distance and angles between pairs of points on the boundary of an object for recognition.

In this work, pairs of edges (or more generally, any region with well-defined and coherent orientation) are used instead of pairs of points (Figure 5.2). Pairs of edges are more distinctive than pairs of points, since they have relative orientation and size. And if used carefully during matching, they contain redundant information about the transformation between image and model. A disadvantage is that edges are subject to occlusion, and edges/regions found automatically many be incomplete or broken into segments. But in all but the most trivial objects, there are many pairs of edges, so this approach is at least not doomed from the start.

The orientation filter developed earlier is applied to images, and a simple region growing algorithm divides the image into sets of contiguous pixels with coherent orientation. For real-time operation, adaptive thresholding on the minimum size of such regions is applied, so that the number of regions is bounded, independent of scene complexity. In "model" (training) views, every pair of regions belonging to the object is considered exhaustively, and

entered into a hash table, indexed by relative angle, relative position, and the color at sample points between the regions (if inside the object boundary).

A useful extension of geometric hashing is coherency, where each match implies a particular transformation, and only "coherent" matches are aggregated (for example, see Lamiroy and Gros (1996)). This could be applied in the present instance. For speed, an abbreviated version is used here, where we filter by the centroid location each match implies for the object (this is information we would like to have anyway). There is no coherence checking by scale and orientation at the matching stage. This procedure means that we perform object localization simultaneously with matching.

Oriented regions are relatively sparse. Experiments showed that on a fast machine (800MHz) and at low resolution (128×128) it is possible to use triplets of regions as features at close to real-time. These can be very distinctive, and very redundant, and non-trivial objects have very very many possible triplets. But the frame-rate was just too slow (approximately 1 Hz) to be worth using here.

At the other extreme, another possibility would be just to use single edges. But they are not very distinctive, and sampling colors at an edge (generally a high-variance area) is problematic.

5.3 Details of matching

The basic feature used is pairs of oriented regions, containing the following information:

- ▷ Angle between the regions. The angle associated with each region is the mean response to the orientation filter in that region, and not the principle axis of the region itself, although these two measures will generally be highly correlated.
- ▷ Positioning of oriented regions relative to each other and to their projected intersection point (normalized for scale). The position associated with a region is the centroid of its silhouette.
- ▷ The color of sample points between the regions. Three points are sampled at quarter-length intervals along the line between the two region centroids, and their colors are compared. Color judgements are normalized for luminance and quantized to just a few bits.

We aim to have about the same number of regions as there are pixels in a row or column of an image, so the square is on the order of the number of pixels in the image, and hence on a scale that the computational architecture is designed to work with comfortably in real-time.

Now that each basic feature can make a prediction of where the object center should be, we can simply accumulate this and see if there is a convergence of evidence. This automatically gives high quality locations and scales the target may plausibly be at.

Segmentation of the object is then possible by seeing which regions contributed to locating the object. At this point, it is useful to find the consensus scale and eliminate matches at other scales.

Once we have proposed regions for the object, we can use any of the methods from the previous chapter to confirm the presence of the object.

The histograms matched against are constructed by merging features from all the training views available. With these histograms, we are modeling both the noise in sensing and the variation in the appearance of the object. This index should be invariant to translation and in-place rotation; some robustness to the precise boundaries of the segmented region. Clearly, we could be unlucky, and small changes could push us over a histogram bin boundary. Could use smoothing, or multiple samples.

For multiple targets, can remove the intersections between histograms (or assign to the target for which a feature is more frequent). Some objects simply have more oriented regions than others, and so may have a greater response to random background. Hence the importance of an independent confirmation method.

5.4 Searching for a synthetic object in a synthetic scene

As a simple example of how this all works, consider the test case shown in Figure 5.3. The system is presented with a model view of the circle, and the test image. For simplicity, the model view in this case is a centered view of the object by itself, so no segmentation is required. The processing on the model and test image is the same – first the orientation filter is applied, and then regions of coherent orientation are detected. For the circle, these regions will be small fragments around its perimeter. For the straight edges in the test image, these will be long. So finding the circle reduces to locating a region where there are edge fragments at diverse angles to each other, and with the distance between them generally large with respect to their own size. Even without using color, this is quite sufficient for a good localization in this case. The perimeter of the circle can be estimated by looking at the edges that contribute to the peak in match strength. The algorithm works equally well on an image of many circles with one square.

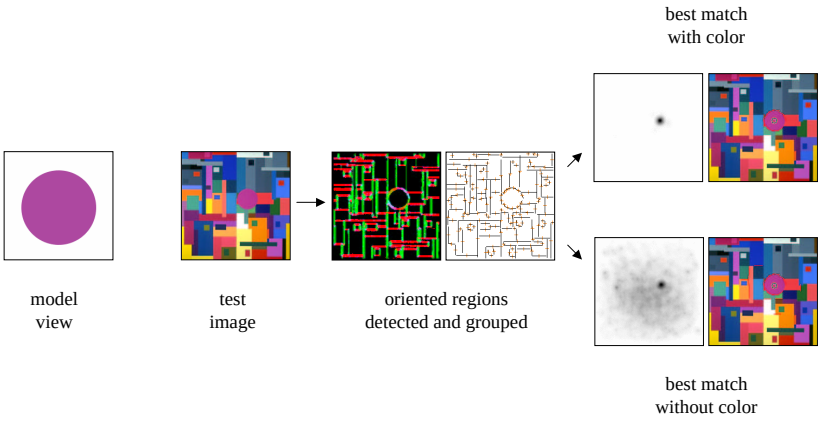


Figure 5.3: A simple example of object localization: finding the circle in a Mondrian.

5.5 Searching for real objects in synthetic scenes

In Figure 5.4, we take a single instance of an object found through poking, and search for it in a synthetic image containing an abstract version of it along with various distractors. The algorithm picks out the best match, and lets us rank the distractors in order of salience. It is clear that a yellow square with anything in it is a good match, and having the internal purple square adds another boost. The closest distractor is a yellow square with a purple square inside it, rotated by 45°. Figure 5.5 shows another example. The object in question is a cube with a green face containing a red triangle. When presented with an image containing numerous variations on this theme, the most reasonable match (in the author's judgement) is selected.

5.6 Recognizing real objects in real images

Figure 5.6 shows examples of the cube being resegmented in real images. Testing on a set of 400 images of four objects (about 100 each) being poked by the robot, with half the images used for training, and half for testing, gives a recognition error rate of about 2%, with a median localization error of 4.2 pixels in a 128×128 image (as determined by comparing with the center of the segmented region given from automatic segmentation). By segmenting the image by grouping the regions implicated in locating object, and filling in, a median of 83.5% of the object is recovered, and 14.5% of the background is mistakenly included (again, determined by comparison with the results of

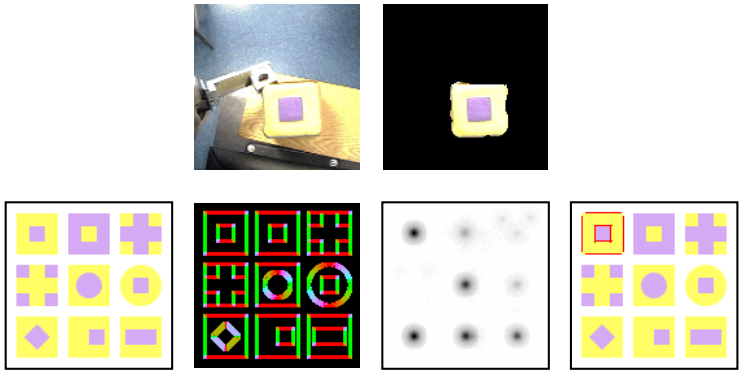


Figure 5.4: Looking for the best match to a cube found through poking in an image of a synthetic version of the cube along with a set of distractors. The superimposed lines on the rightmost image in the bottom row indicate the detected position of the object and the edges implicated. The image on its immediate left shows the strength of evidence for the object across the entire image, which lets us rank the distractors in order of attractiveness.

automatic segmentation).

5.7 Dealing with multiple objects simultaneously

There is nothing to stop us dealing with multiple matches in the same image, as shown in Figure 5.7. This is not in fact important for the robotic implementation, since it uses a foveated approach where objects are viewed sequentially.

5.8 Online training

In geometric hashing, the procedure applied to an image at recognition time is essentially identical to the procedure applied at training time. We can make use of that fact to integrate training into a fully online system, allowing behavior such as that shown in Figure 5.10, where a previously unknown object can be segmented through active segmentation and then immediately localized and recognized in future interaction.

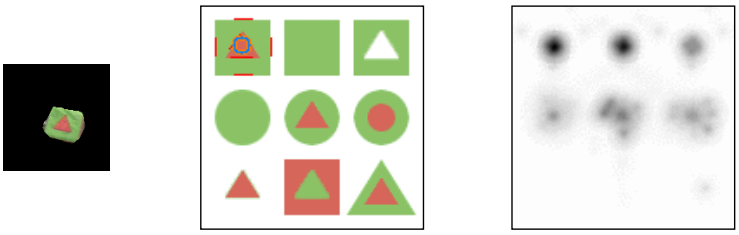


Figure 5.5: If the recognition system is trained on real images of a green cube (typical segmentation shown on left), and presented with a synthetic version of the cube along with a set of distractors (middle), we can evaluate what features are used for recognition. The superimposed circle and lines indicate the detected position of the object and the edges implicated. The image on the right shows the strength of evidence for the object across the entire image, which lets us rank the distractors in order of attractiveness. In this case, the most prominent feature used in recognition is the outer green square.

5.9 Extracting an object prototype

The model developed above is good for localization and recognition, but is difficult to visualize directly. It is useful to have a less powerful but more easily viewable model to get a quick sense of how well things are working. This section develops a simple procedure for doing so.

Since every view of an object has a segmentation mask, we can try to align these masks, and then average the views to get a prototype for the object. First the masks are centered. Then they are rotated to minimize the moment in the vertical direction – if the object's profile is asymmetric, this will normalize the object to two possible orientations. An iterative approach is them used to flip the masks such that the corresponding object views are as similar as possible. Finally, the resulting transformed views are averaged, as shown in Figure 5.11. The average is typically blurry, and there may be some aspect of pose that wasn't normalized (see for example the green cube in Figure 5.11). The final step is to find the segmentation that best matches the average. This gets us back to a particular view of the object, but one that is as representative as possible. This gives us a good feel for how good the segmentations are. The results here show, for example, that all four objects are well-segmented. The bottle seems to be systematically losing its cap, and the small strip below the label. This suggests that there may be problems when there is a strong internal edge slightly inside the boundary – the segmentation procedure may switch to using them to minimize perimeter length. This should be fixable, but the point is that it is much easier to see that kind of effect in this repre-

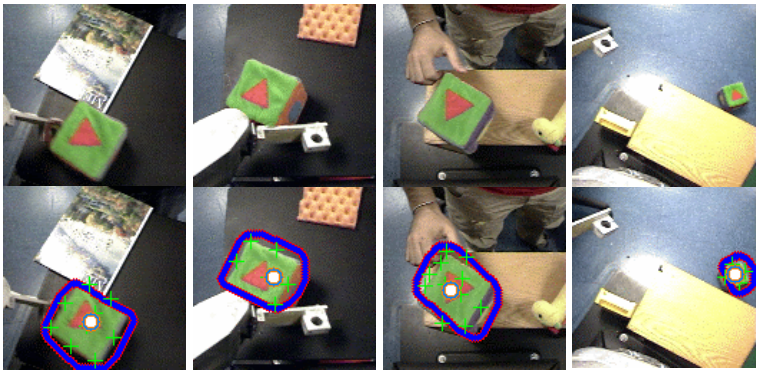


Figure 5.6: The cube being recognized, localized, and segmented in real images. The image in the first column is one the system was trained on. The image in the remain columns are test images. Note the scale and orientation invariance demonstrated in the final image.

sentation than in the earlier one, even though it would be much less useful for localization purposes.

5.10 Comparing segmentations

To build up a good object model, it is useful to combine information across multiple segmentations. But different segmentations may come from different objects. Hence the segmentations need to be compared and clustered. The images are coming from online experience, and so have timestamps associated with them. Images close in time are likely to be from the same object. However, it would be useful to be able to pool data across different training episodes. Hence other cues such as color, boundary shape, and oriented features are important.

A variety of measures were evaluated. Comparing segmented views of objects is a relatively easy problem, and so much work has been done on it that it seemed unnecessary to spend time on this. So a simple measure (color histogram comparison) was adopted. Clustering happens both online and offline. The online implementation is optimized for speed, the offline method is optimized for accuracy. As offline updates become available, they replace the online object models.

- ▷ **Clustering by color histogram:** A classic Swain and Ballard (1991) style implementation was used, giving a recognition accuracy of 98.6%

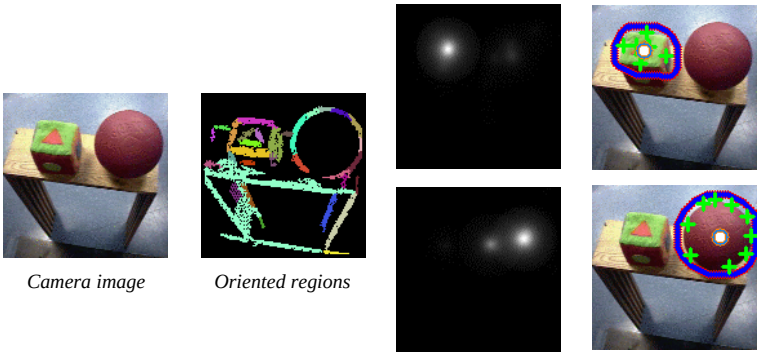


Figure 5.7: It is possible to deal with multiple objects in the same image.

on a set of 420 images of the four objects shown in Figure 5.11, measured using leave-one-out cross validation.

- ▷ **Clustering by orientation histogram:** If color information was replaced with a histogram of orientation angles detected within the object, then 88.6% accuracy was achieved (See Figure 5.8). Using orientation is complicated by the fact that histograms have an a degree of freedom as the object rotates, so when comparing two histograms they need to be aligned first. This is slow compared to using color.
- ▷ **Clustering by boundary shape** Using shape information based on the first four Hu moments of the segmented boundaries, an accuracy of 87.6% was achieved.
- ▷ **Clustering by behavior** In theory, objects could be clustered based on how the move when poked. Inaccuracies in motor control made this impractical. However, this topic is revisited in Chapter 7 where aggregate, statistical measures of object behavior are shown to indeed be measurable.

5.11 Stabilized perceptual interface

One problem with trying to continually refine models for recognizing objects is that changing models could confuse any modules downstream that refer to these models. To deal with this a *stabilized interface* approach is used. For each object (or word, see Chapter 9) that can be recognized, a 'feature line' is allocated. The contract between the recognizing module and the rest of that

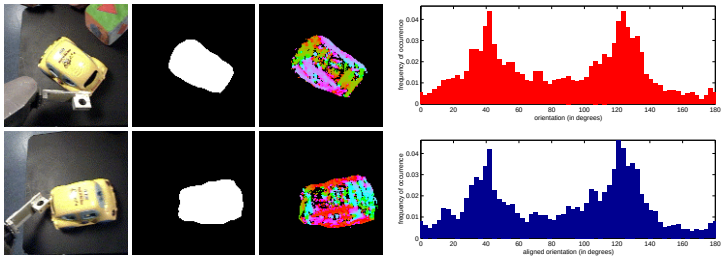


Figure 5.8: Comparing object segmentations using orientation histograms.

control system is that the 'semantics' of that feature line will be preserved as much as possible – it will respond to the same situations in future as it did in the past – except for attempts to refine or purify the semantics (so that it is less affected by noise, for example, or responds to the same basic property in an extended range of situations). Offline updates are made initially without any regard to the contract so that off-the-shelf clustering algorithms can be used; then as a last step models are compared with previous models and aligned appropriately.

5.12 Completion and illusory contours

If there are not too many of them, the object recognition process will consider completing breaks in edges, if there are edges with the same orientation which, if extended, overlap with each other. The main purpose of this is to compensate for limitations of region grouping, which will sometimes erroneously split an oriented region into two fragments. As a byproduct, simple illusory contours can be dealt with, such as the Kanizsa triangle shown in Figure 5.9.

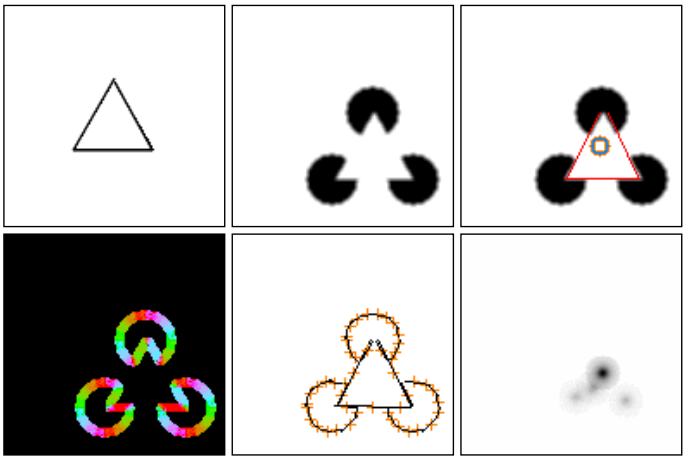


Figure 5.9: Breaks in edges are considered for completion during the object recognition process. Hence certain illusory contours such as the triangle in the right of this figure possesses, can be recognized.

Figure 5.10: This figure shows stills from a three-minute interaction with Cog. The area of the first frame highlighted with a square shows the state of the robot – the left box gives the view from the robot's camera, the right shows an image it associates with the current view. Initially the robot is not familiar with any objects, so the right box is empty. It is presented with the cube, and pokes it (first frame). Then, if shown the cube again, it recognizes it (this recognition is evidenced by showing an image recorded from when the object was poked). If the cube is turned to another side, the robot no longer recognizes it (third frame). If that side of the cube is presented to the robot to poke (fourth frame), if can then recognize it (fifth frame) and differentiate if from the green side (sixth frame). If it confuses another object with what it has already seen, such as the ball in the seventh frame, this is easily to fix by poking (eighth, ninth frames). The final three frames show the invariance of the recognition system to scale.

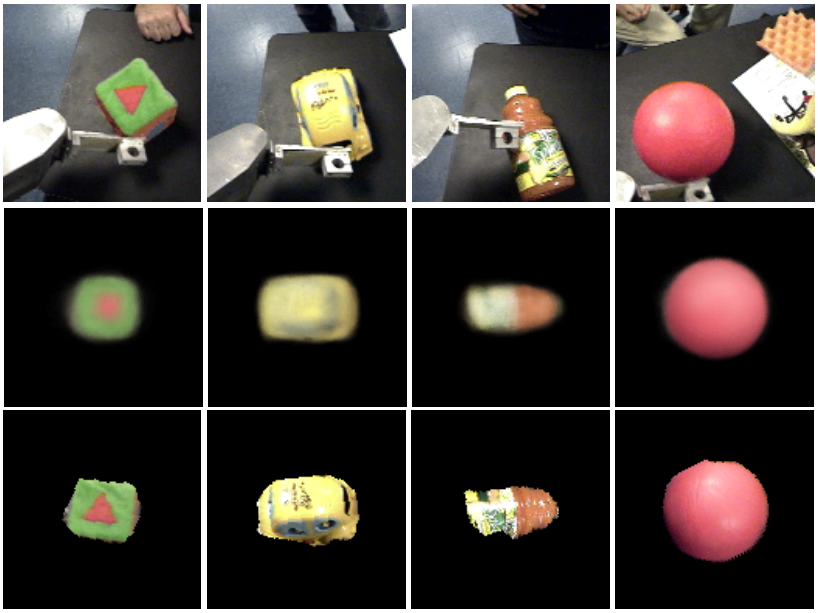


Figure 5.11: The top row shows the four objects used in this experiment, seen from the robot's perspective. The middle row shows prototypes derived for those objects using a naïve alignment procedure. None of the prototypes contain any part of the robot's manipulator, or the environment. These prototypes are used to find the best available segmentations of the objects (bottom row).

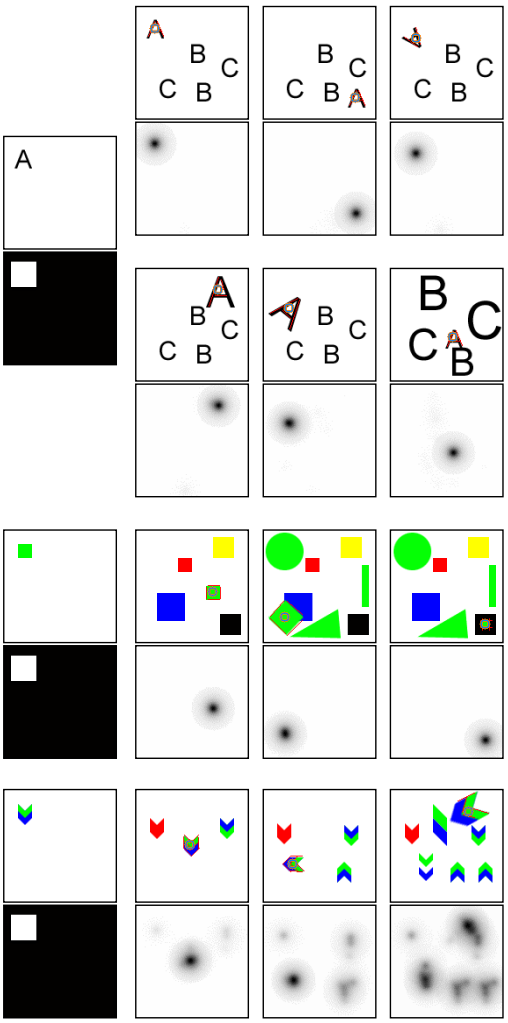


Figure 5.12: More localization examples. The left-most column shows the object prototype and mask for training. The top half of the figure shows the letter A being located at various positions, scales, and orientations (B and C work fine too, but letters that are physical supersets of each other with the same center, such as C and O, cannot be distinguished). The small circle indicates the best matching location, and the full map of responses is given underneath each image. The lower half of the figure shows the same algorithm working on examples of a different class of object, colored shapes: a simple square, and a chevron.

CHAPTER 6

Reaching out: discovering one's own (and other) manipulators

> *He'd never bitten a hand that fed him. After all, this made it so much harder for the hand to feed you tomorrow.* (Pratchett, 1999)

In a sense, poking provides the robot with an operational definition of what objects are by giving it an effective procedure for learning about them. It is not perfect – for example, the robot is effectively blind to objects that are too small or too large – but for objects at an appropriate scale for manipulation, it works well. Once the robot is familiar with a set of such objects, we can go further and provide an operational definition of a *manipulator* as something that acts upon these objects. This chapter develops and effective procedure for grounding this definition.

To get training images of the manipulator, we need to find an opportunity when we can both segment it from the background and be sure that the segmented region is in fact the manipulator. Without constraining the environment, or having a prior training period in which a hand-eye mapping was trained, this is quite hard to do. However, there is one ideal opportunity – the moments before a poking event. This is a fairly narrow time window, when the robot is fixating and actively trying to move the arm into view. There is also an independent measure of whether the arm is in fact in view and whether everything is proceeding smoothly, which is the contact detection algorithm.

Together, these can identify a short period of time in which the manipulator is very likely to be visible and moving across the field of view.

6.1 Hand to eye coordination

A robot has privileged knowledge of the state of its arms. Hence it can in principle predict from proprioceptive feedback where the arms will appear in the images from its cameras. Learning the mapping from joint angles to retinotopic coordinates is a favorite task in robotics (Fitzpatrick and Metta, 2002; Marjanović et al., 1996; Metta et al., 1999). Detection of the endpoint of the manipulator during training is made trivial by either giving it a special color, or by shaking it repeatedly. Once the mapping is learned, this simplification is no longer necessary, since the mapping does not depend on visual appearance. But what if we did want to learn a mapping that depends on appearance? For example, it would be useful if the robot could independently estimate the location of the arm from visual evidence rather than motor feedback, so that it could do precise closed-loop visually-guided control, rather than just 'blind' open-loop reaching. Now if we make the endpoint obvious using color or repeated motion in order to detect the manipulator, we must be careful that we can actually move away from that constraint after training.

The solution adopted is to identify a special situation in which the robot's arm can be identified in the visual field under its normal behavior. Consider the basic poking behavior introduced in Chapter 3. The visual collision detection mechanism developed in that chapter operates without any appearance model of the arm. When it does detect a collision near the point of fixation towards which the arm is being driven, that collision is very likely to be between the arm and an object. In Chapter 3 the motion caused by this collision was used to segment the object, with any motion regions that appeared to originate before the collision being discarded as being due to the arm, its shadow, or background motion. We can turn this reasoning around and try to segment the object that was moving before the collision. These segmentations are likely to contain the arm. As was shown in Chapter 5, if sufficiently good segmentations can be collected, then they can be refined – they don't need to be perfect.

If behaviors like poking are possible using open-loop reaching, is there any real reason to develop visually-guided reaching? As will be seen Chapter 7, although open-loop poking is fine for segmentation purposes, it leaves a lot to be desired when actually trying to move an object in a controlled way.

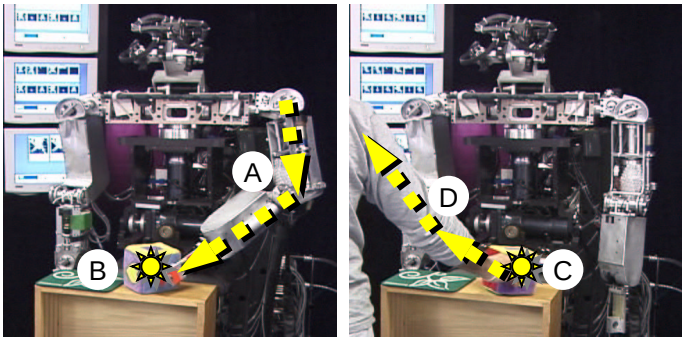


Figure 6.1: On the left, the robot establishes a causal connection between commanded motion and its own manipulator (A), and then probes its manipulator's effect on an object (B). The object then serves as a literal "point of contact" (C) to link robot manipulation with human manipulation (on the right, D).

6.2 Objects as intermediaries

There is another motivation behind the choice of the poking behavior as the vehicle for detecting the manipulator. Clearly contact with objects is not the only possible way to locate the robot's arm, if we are willing to construct some special training behavior, such as a simple analogue of a human infant's hand-regard behavior. But it does have the advantage that objects can also serve as a point of contact between robot and human (Kozima et al., 2002). This thesis has shown that a robot can use its manipulator to familiarize itself with objects by poking them; if the robot then observes those known objects behaving as if they are being poked – but without it actually poking them itself – then it can reasonably deduce that they are being acted on by some entity like itself (see Figure 6.1). For this to work, the robot must be able to perceive poking carried out by another. This is not easy to do in general, since the robot's gaze is unlikely to be directed at the appropriate location. But once it is familiar with an object, it can maintain fixation on it for a long period of time. Or if it sees a reachable object, familiar or unfamiliar, and begins to poke it, it will also fixate. It is simple to add a behavior that suppresses poking if unexpected motion is detected around the object (not due to the robot's own movement). This means that a human, noticing that the robot is preparing to poke an object, can take over and poke it themselves. Then the same machinery developed for active segmentation to operate when a foreign manipulator (such as the human hand) pokes the fixated object. Of

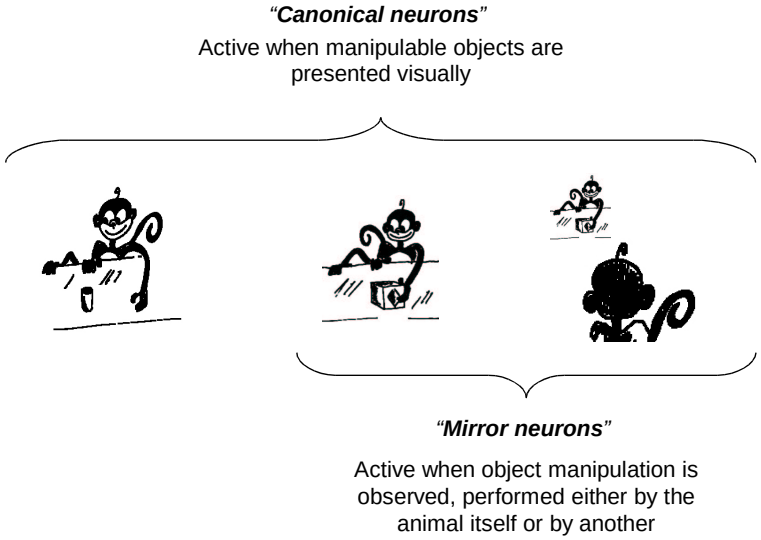


Figure 6.2: Canonical neurons are active when a manipulable object is observed, but this is not sufficient to activate mirror neurons. Mirror neurons are active only when goal-directed manipulation of an object is observed. This manipulation may be performed either by the animal itself, or by another (Gallese et al., 1996). These classes of neuron have been observed in area F5 of both monkeys and humans. (Monkey drawings by Giorgio Metta).

course the robot can easily distinguish segmentations produced using its own arm from that of others simply by checking whether it was commanding its arm to move towards the target at the time. In this way it can also build up a model of foreign manipulators belong to others in its environment.

6.3 Canonical and mirror neurons

Is there any strong reason to identify the representation of the robot's own arm with the arms of others from the very beginning? Much of vision research is motivated by hints from biology. For primates, many neurons are concerned with both vision and motor control. For example, Fogassi et al. (1996) and Graziano et al. (1997) observed neurons that have receptive fields in somatosensory, visual, and motor domains in area F4. Motor information appears to be used to keep the somatosensory and visual receptive fields anchored to a particular part of the body, such as the forearm, as the body

moves. So-called 'canonical' neurons have been located in area F5 which respond when the host acts upon an object in a particular way (such as grasping it with a precision grip), or when it simply fixates the object (Jeannerod, 1997). These responses are very specific to the type of object and type of action, and so have been interpreted as being the neural analogue of the affordances of Gibson (Gibson, 1977). Affordances are discussed in detail in Chapter 7; briefly, they are simply possible actions that a particular actor can apply to a particular object. A related class of neurons in F5 called 'canonical' neurons have also been identified in primates (including humans) which respond when the host is performing a particular action (such as grasping) or when the host observes someone else performing that same action (Gallese et al., 1996). Again this response is very specific and hard to fool – if a neuron is selective for a particular type of grasp, it does not respond if a tool such as a pliers in used instead. These neurons have been interpreted as a possible basis for imitative behavior, and perhaps even language (Rizzolatti and Arbib, 1998). The existence of these neurons are a motivation for this work.

6.4 Implementation details

The manipulator can be segmented by hypothesizing that it moves towards the object at a constant velocity in the period immediately preceding the moment of contact. Estimating the velocity from the gross apparent motion allows the segmentation problem to be expressed in the form introduced in Chapter 3, where the foreground is now taken to be regions moving at the desired velocity, and the background is everything else. The apparent motion is computed by finding at each frame the translation that best aligns the differences between successive frames (rotation is ignored on this short timescale). Each pixel is assigned either to the stationery background or to a layer moving at the estimated rate. Typical results of this segmentation procedure are shown in Figure 6.3 for both the robot's arm and a human hand. Although it isn't relevant to the current discussion, segmentation of the object happens in the same way after human action as it does after robot action (Figure 6.4 shows an example).

6.5 Modeling the manipulator

As a first test to make sure the segmentation data was of usable quality, it was passed through the same alignment and averaging procedure described for objects in Chapter 5. The only modification made was that segmentation masks were now right-aligned rather than center-aligned, since differing lengths of the manipulator can be in view. This builds in the assumption that

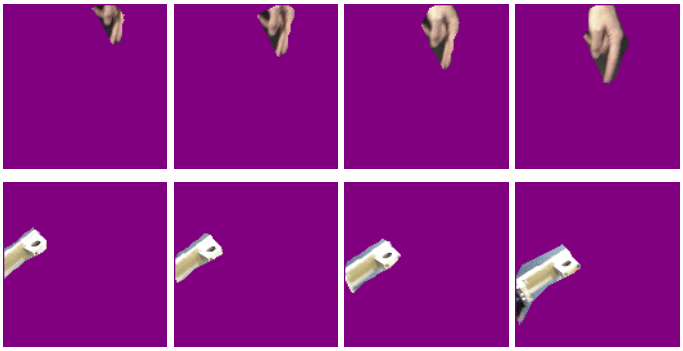


Figure 6.3: Experiments on segmenting the robot arm, or a human arm poking an object. The segmentation is performed by working backwards from the point of collision the object, which occurs in the frame immediately following the ones shown.

the manipulator is long and thin. Figure 6.5 shows the results of alignment. The segmentation that best matches the averaged prototype is a good segmentation. So the basic goal of the procedure, to acquire good images of the manipulator, seems to be reasonably met.

The segmentations can be passed directly to the object localization system developed in Chapter 5, with one modification. Again, the 'center' of the manipulator is not well-defined, we are better off working with its endpoint. This could be detected from where the manipulator strikes the object during the segmentation procedure, but as a shortcut the endpoint was simply defined as the rightmost point in the object (this works fine since the robot only pokes with its left arm). Typical localization results are shown in Figure 6.6.

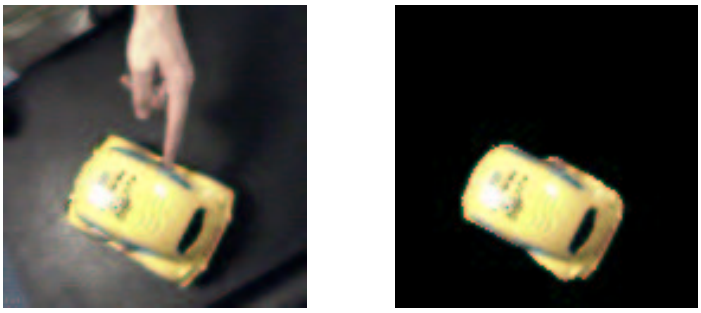


Figure 6.4: The segmentation algorithm will work for human poking operations, if the robot is fixating the object the human pokes. The robot can be made to fixate objects by bringing its attention to it by any of the means discussed in Chapter 8.

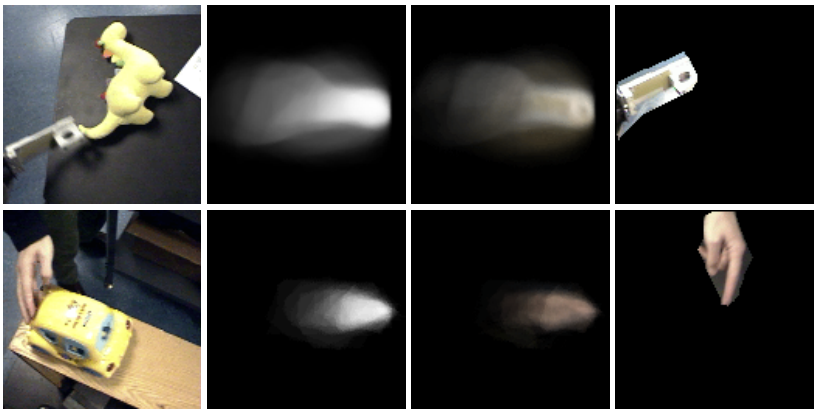


Figure 6.5: The robot manipulator (top left) was automatically segmented during 20 poking sequences. The segmentations were aligned and averaged, giving the mask and appearance shown in the adjacent images. The best matching view is shown on the top right. A similar result for the human hand is shown on the bottom, based on much less data (5 poking sequences, hands of two individuals).

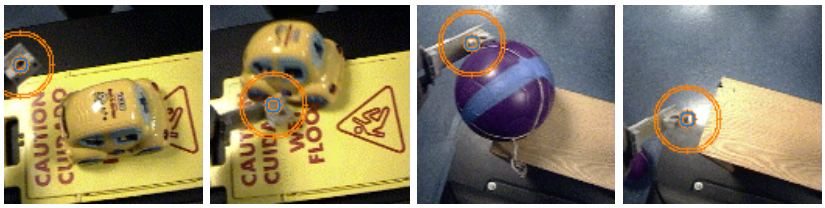


Figure 6.6: After training, the endpoint of the robot's arm can be reliably detected when it is in view (as indicated by the orange circle), despite variations in distance to the arm and how much of the arm is in view.

CHAPTER 7

Rock and roll: exploring and exploiting an object affordance

> *[The waterfall] was the second highest anywhere on the Disc and had been discovered in the Year of the Revolving Crab by the noted explorer Guy de Yoyo. Of course, lots of dwarfs, trolls, native people, trappers, hunters, and the merely badly lost had discovered it on an almost daily basis for thousands of years. But they weren't explorers and didn't count.* (Pratchett, 1991b)

In the end what matters in life and robotics is action, and perception should reflect that priority. Perception can be seen as "basically an implicit preparation to respond" (Sperry, 1952). This chapter introduces an approach to perception that is *explicitly* about preparation to respond. The perceptual system is assigned the task of continually preparing a set of actions that are possible in the robot's current situation, and which simply need to be selected and activated to take effect. This approach has similarities with Gibson's notion of affordances (Gibson, 1977), which is reviewed.

7.1 What are affordances?

Affordances are possibilities for action. If a creature can perform some action on an object, then the object is said to afford that action, and that action is an affordance of the object. For example, a cup affords grasping and drinking.

The idea of affordance is actor-specific; a leaf might afford support to an ant but not an elephant. The existence of an affordance depends only on whether the creature can perform the appropriate actions, and does not depend on the ability of the creature to perceive it. Other authors have used the term in different ways. Those concerned with interface design, such as the Human-Computer Interface community, often use both perception and action as the defining characteristics of the term – see McGrenere and Ho (2000) for a review. We will take "perceived affordances" to refer to the actions a creature believes are possible on an object, which are potentially distinct from the "true affordances" that are physically realizable.

In Gibson's work, there is an implication that affordances can be perceived "directly" and are in some sense "picked up" from the environment – as opposed to being inferred. This is not a particularly helpful notion for robotics, and although there is a significant literature on the notion of direct perception, it will not be reviewed here (see Hurley (2001) for a discussion). Gibson did make good points about vision being easier when done dynamically from a moving perspective, ideas that cropped up later as active/animate vision. Gibson pointed out the power of optic flow information, which this thesis has benefited from (and in other collaborative work even more (Fitzpatrick and Metta, 2002)).

7.2 Why think about affordances?

In robotics, possibilities for action are captured concisely in a robot's configuration space. A configuration space contains all the parameters (e.g. joint angles) necessary to uniquely specify the robot's physical state. Actions correspond to trajectories in the configuration space. So why do we need another way to think about the space of possible actions?

Configuration space is very useful for planning the details of motor control, such as getting from one point to another without trying to move through any impossible joint angles. If there are complicated constraints on action then this tactical level of analysis is unavoidable. However, at the strategic level, it isn't as helpful. When the robot is deciding what it should do now, joint angle trajectories are the wrong level of abstraction. One choice would be to switch into a space with a representation of goals and possible operators, and do planning, and then later translate operators back into motor actions using plan execution. This involves a significant architectural commitment. It is useful to consider if there are alternatives that don't involve such a dramatic "phase change" between motor control and perception. Affordances offer such an alternative. If there is a predominantly bottom-up system assessing what is possible in the robot's current situation, then it can prepare the appro-

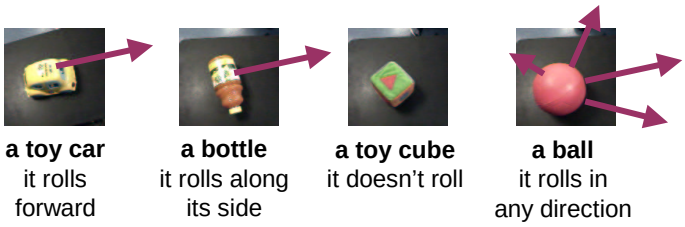


Figure 7.1: Different objects roll in different ways. A toy car rolls forward, a bottle rolls on its side, a ball rolls in any direction, and a cube doesn't roll easily at all.

priate control parameters for the available actions, and describe the actions in a very low-bandwidth way relative to this – with all the awkward details suppressed. Is this different from planning, picking an action, and then assessing what parameters are appropriate to carry it out? Not in principle, but in practice it could significantly simplify the decisions that need to be made.

Configuration space ideas do have the benefit of being formalized and clear, unlike affordances. We could define an "affordance space" as the set of control parameters output by perception so that initiated actions are channelled appropriately, and then a set of action flags specifying which actions seem possible. For example, an affordance-based perceptual system for a mobile robot might chose to signal "turn" as a possible action to its behavior system, while setting up appropriate control parameters to achieve a good turn or to continue on straight. If all the robot does is navigate then there is not much benefit to this; but if the robot has a wide vocabulary of actions then this may be a useful simplification. The danger of using a weaker perception system that makes minimal judgements itself is that it will add delay, and potentially leave decisions in the hands of a less well-informed module. Of course, not everything is evident from perception, so integration is still important.

7.3 Exploring an affordance

The examples of affordances that are most commonly discussed include different kinds of grasping, twisting, or chewing. All of these require quite sophisticated motor control. As a starting point, it seemed more sensible to choose actions that have all the properties of an affordance, but have a lower cost of entry in terms of dexterity. The author identified object rolling as an excellent candidate. Only certain objects roll well, and to make them roll requires matching the robot's action to the object's pose in an intelligent

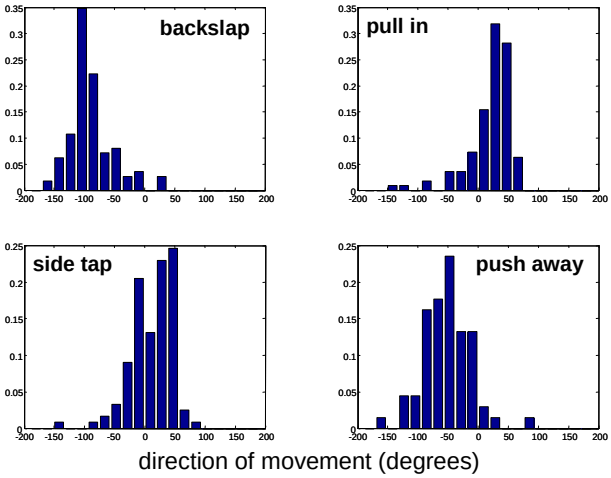


Figure 7.2: Histogram of the direction of movement of object for each possible poking action. For each of the four plots the abscissa is the direction of motion of the object where the 0° direction is parallel to the x axis, and -90° to the y axis. The ordinate is the empirical probability distribution of the direction of motion of the objects.

manner. For example, Figure 7.1 shows four objects that have quite distinct properties in terms of a "rolling affordance". The rolling affordance is perfectly within reach of the robot, given the capabilities already developed. It can poke an object from different directions, and it can locate familiar objects and recognize their identity. Active segmentation played two roles in this experiment: collecting data for later object recognition and localization, and providing a good segmentation for tracking the motion of the object after contact. Chronologically, this experiment was performed before the techniques for tracking, recognition, and localization described elsewhere in this thesis were fully developed, so simpler methods were used (color histogram backprojection for localization and recognition, optic flow based tracking over a small number of frames). This system was developed in collaboration with Giorgio Metta.

We designed two experiments that use poking and the visual segmentation described in Chapter 3 to move an object on the basis of the rolling affordance. In the first experiment the robot poked the set of objects shown in Figure 7.1 (an orange juice bottle, a toy car, a cube, and a colored ball) using one of four possible actions (the motor repertoire). Actions are labelled for convenience as back-slap, side-tap, pull-in, and push-away. These actions

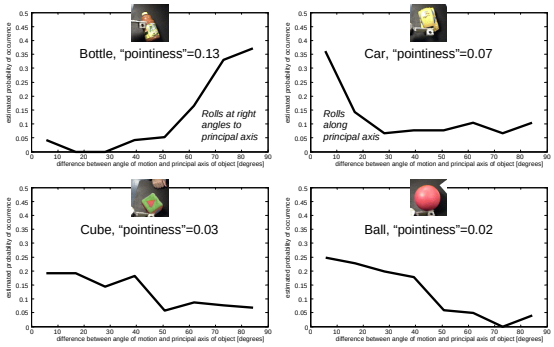


Figure 7.3: Probability of observing a roll along a particular direction for the set of four objects used in Cog's experiments. Abscissae represent the difference between the principal axis of the object and the observed direction of movement. Ordinates are the estimated probability. The principal axis is computed using the second Hu moment of the object's silhouette (Hu, 1962). The "pointiness" or anisotropy of the silhouette is also measured from a higher order moment; this is low when the object has no well-defined principal axis, as is the case for the cube and the ball. The car and bottle have clear directions in which they tend to roll. In contrast, the cube slides, and the ball rolls, in any direction. These histograms represent the accumulation of many trials, and average over the complicated dynamics of the objects and the robot's arm to capture an overall trend that is simple enough for the robot to actually exploit.

correspond to different patterns of poking. In a side-tap, the robot sweeps its arm in across the field of view. In a back-slap, the robot first raises its arm and draws it in to its torso, then sweeps outwards. Normally these actions are used interchangeably and at random during poking, as the segmentation algorithm is agnostic about the source of object motion (see for example Figure 3.11). The toy car and the bottle tend to roll along a definite direction with respect to their principal axis. The car rolls along its principal axis, and the bottle rolls orthogonal to it. The cube doesn't really roll because of its shape. The ball rolls, but in any direction. Shape information can be extracted from the segmentation produced by poking, so these relationships could in principle be learned – and that is the goal of this experiment.

The robot poked the set of objects shown in Figure 7.1 many times (approximately 100 each), along with other distractors. The segmented views were clustered based on their color histogram. For each poking episode, shape statistics were gathered at the point of impact, and the overall trans-

lational motion of the object was tracked for a dozen frames after impact. Over all poking events (470 in all) the gross translation caused by poking was computed as a function of the type of poking applied (back-slip, side-tap, pull-in, push-away), as shown in Figure 7.2. This is necessary since the effect that the poking fixed action pattern has is not known to the robot's perceptual system; this procedure recovers the effect (and also reveals that the motor control on Cog's arm is very erratic). Using this procedure, the robot automatically learns that poking from the left causes the object to slide/roll to the right, as a general rule. A similar consideration applies to the other actions. Next, object-specific models are built up relating the effect of object orientation on the translation that occurs during poking. Figure 7.3 shows the estimated probability of observing each of the objects rolling along a particular direction with respect to its principal axis. Here the peculiar properties of the car and bottle reveal themselves as a "preferred direction" of rolling. The ball and cube do not have such a preference. At the end of the learning procedure the robot has built a representation of each object in terms of:

- ▷ Pictorial information in the form of color histograms, following (Swain and Ballard, 1991).
- ▷ Shape information in the form of a measure of the average size of the object, an index of the elongation of the object with respect to its principal axis, and a set of Hu moments (Hu, 1962).
- ▷ Detailed histograms of the displacement of the object with respect to its initial orientation given that a particular motor primitive was used.
- ▷ The summary histograms shown in Figure 7.3 which capture the overall response of each object to poking.

After the training stage, if one of the known objects is presented to Cog, the object is recognized, localized and its orientation estimated (from its principal axis). Recognition and localization are based on the same color histogram procedure used during training (Swain and Ballard, 1991). Cog then uses its understanding of the affordance of the object (Figure 7.3) and of the geometry of poking to make the object roll. The whole localization procedure has an error between 10° and 25° which which is perfectly acceptable given the coarseness of the motor control We performed a simple qualitative test of the overall performance of the robot. Out of 100 trials the robot made 15 mistakes. A trial was classified as "mistaken" if the robot failed to poke the object it was presented with in the direction that would make it roll. The judgements of the appropriate direction, and whether the robot succeeded in actually achieving it, were made by external observation of the behavior of the robot. Twelve of the mistakes were due to imprecise control – for example the manipulator sometimes moved excessively quickly and shoved the object outside the field of view. The three remaining errors were genuine mistakes

due to misinterpretation of the object position/orientation. Another potential mistake that could occur is if the robot misidentifies an object – and, for example, believes it sees a bottle when it in fact sees a car. Then the robot will poke the object the wrong way even if it correctly determines the object's position and orientation.

7.4 Mimicry application

With the knowledge about objects collected in the previous experiment we can then set up a second experiment where the robot observes a human performing an action on the same set of objects, and then mimics it. In fact, the same visual processing used for analyzing a robot-generated action can be used in this situation also, to detect contact and segment the object from the human arm, as described in Chapter 6. The robot identifies the action observed with respect to its own motor vocabulary. This is done by comparing the displacement of the object with the four possible actions, as characterized in Figure 7.2, and choosing the action whose effects are closer to the observed displacement. This procedure is orders of magnitude simpler than trying to completely characterize the action in terms of the observed kinematics of the movement.

The robot can then mimic the observed behavior of the human if it sees the same object again. The angle between the preferred direction of motion of the object (as characterized in Figure 7.3) and the observed displacement is measured. During mimicry the object is localized as in the previous experiment and the robot picks the motor action which is most likely to produce the same observed angle relative to the object. If, for example, the car was poked at right angle with respect to its principal axis Cog would mimic the action by poking the car at right angle, despite the fact that the car's preferred behavior is to move along its principal axis. Examples of observation of poking and generation of mimicry actions are shown in Figures 7.5.

7.5 Conclusions

Describing a problem the right way is an important step to solving it. Affordances provide a means to implement bottom-up influence on the terms in which the current situation is described. For example, in the work described here, if the perceptual system detects that the robot is looking at an object that can roll, then the motor options automatically change. Poking will now automatically push the object in the right direction to make the object roll. The option to strike the object awkwardly is now available – which the robot is

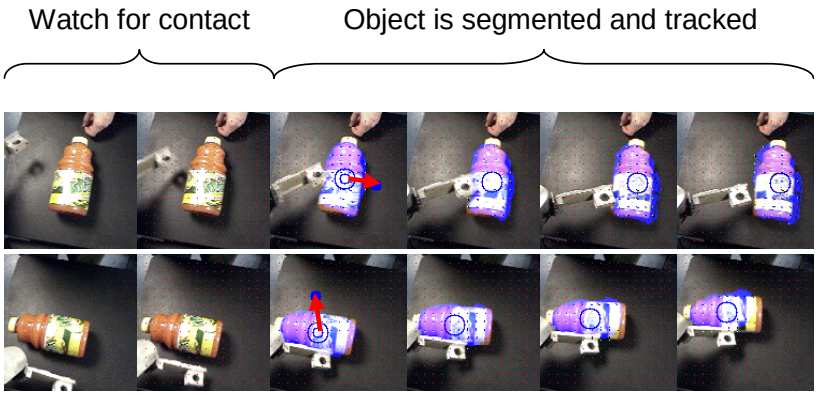


Figure 7.4: Frames around the moment of contact are shown. The object, after segmentation, is tracked for 12 frames using a combination of template matching and optic flow. The big circles represent the tracked position of the bottle in successive frames. The arrow displayed on the frame of contact (3^{rd} from the left) projects from the position at the time of contact and at the 12^{th} frame respectively. In the first sequence, the bottle is presented to the robot at an orientation that makes a side-tap appropriate for rolling, and that is what the robot does. In the second sequence, the car is presented at a different angle. The appropriate action to exploit the affordance and make the bottle roll is now a back-slap.

pretty good at anyway, but now it can do it deliberately, to mimic human action for example. Control in terms of side-taps and back-slaps is still possible of course, but that level of detail is no longer necessary.

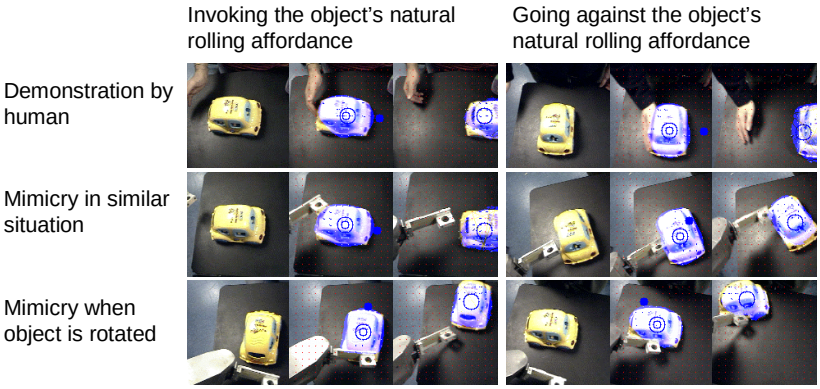


Figure 7.5: A mimicry example using the toy car. The first row shows human demonstration of poking operations, which the robot then mimics. The sequences on the left show the robot mimicking a human exploiting the car's rolling affordance. The sequences on the right show what happens when the human hits the car in a contrary fashion, going against its preferred direction of motion. The robot mimics this "unnatural" action, suppressing its usual behavior of trying to evoke rolling. Mimicry is shown to be independent of the orientation at which the car is presented.

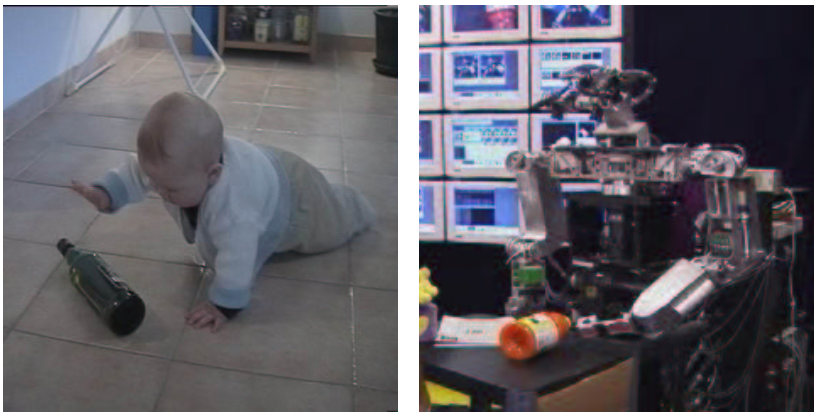


Figure 7.6: You don't have to have high dexterity to explore some properties of objects.

CHAPTER 8

The final frontier: working in space, keeping track of objects

> *"I don't actually think," Ponder Stibbons said gloomily, "that I want to tell the Archchancellor that this machine stops working if we take its fluffy teddy bear away. I just don't think I want to live in that kind of world."*
> *"Er, you could always, you know, sort of say it needs to work with the FTB enabled."* (Pratchett, 1996)

Objects live in 3D space. They move around and change pose. This chapter introduces an array of interconnected methods to track objects and their locations, from low-level visual attention to egocentric maps.

8.1 Overall approach

This work extends and overlaps with previous efforts on Cog and Kismet by Scassellati (2001), Breazeal (2000), and the author. Our work on attention has addressed active vision in a social setting (Breazeal et al., 2001) and a framework for using social cues to the advantage of the robot's vision system (Breazeal and Fitzpatrick, 2000). Breazeal and Scassellati (1999) implemented a model of low level visual attention based on measures of intrinsic salience (brightness, motion etc.) with some behavioral modulation. The

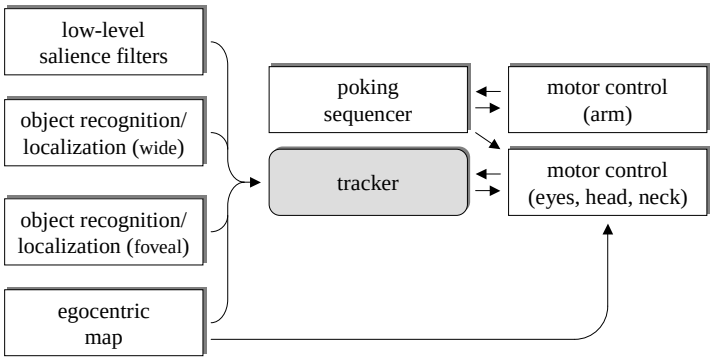


Figure 8.1: Influences on gaze. Object recognition subsumes the low-level salience filters.

author extended that work to introduce a mechanism for controlling persistence – to be able to deliberately hold gaze on an object, and dynamically control the trade-off between being responsive to changes in the environment and being a complete slave to fluctuations in it. This chapter goes further to encompass many more influences on attention (see Figure 8.1), such as object recognition and spatial memory. Figure 8.1) shows the basic architecture of how the robot's gaze is controlled. If influences are coming from modules that operate at different speeds and with different levels of noise versus stability, and the robot's head and eyes were under direct control of these modules, then its behavior will be very erratic indeed. Instead, all visual control of the robot's gaze direction is mediated by a tracking module which operates at the fastest rate possible (30Hz). This means that the robot can respond to fast moving objects even if not all parts of the vision system can react quickly.

Since this work is implemented on a humanoid robot, it is crucial that the robot carefully controls its own gaze to convey the most veridical impression of its state to the human. It helps if the robot moves its eyes in a manner consistent with human eye movement.

Also, spatial 'awareness' is important for tasks, since even for adult humans cognition can often be traded off with physical space (Kirsh, 1995b; Pelz, 1995).

8.2 Human-like gaze

The eye movement primitives implemented on Kismet and Cog are modeled after the human ocular-motor system. The human system is so good at providing a stable percept of the world that we have little intuitive appreciation of

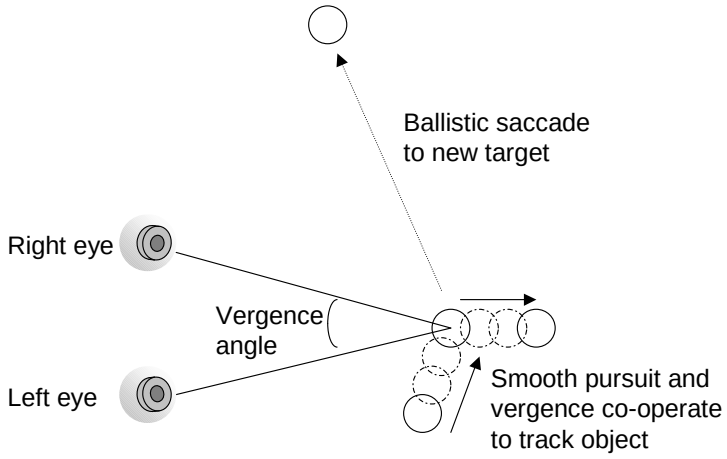


Figure 8.2: Humans exhibit four characteristic types of eye motion. Saccadic movements are high-speed ballistic motions that center a target in the field of view. Smooth pursuit movements are used to track a moving object at low velocities. The vestibulo-ocular and opto-kinetic reflexes act to maintain the angle of gaze as the head and body move through the world. Vergence movements serve to maintain an object in the center of the field of view of both eyes as the object moves in depth. The robotic eye movements are controlled to mirror this pattern.

the physical constraints under which it operates. Humans have foveate vision. The fovea (the center of the retina) has a much higher density of photoreceptors than the periphery. This means that to see an object clearly, humans must move their eyes such that the image of the object falls on the fovea. Human eye movement is not smooth. It is composed of many quick jumps, called saccades, which rapidly re-orient the eye to project a different part of the visual scene onto the fovea. After a saccade, there is typically a period of fixation, during which the eyes are relatively stable. They are by no means stationary, and continue to engage in corrective micro-saccades and other small movements. If the eyes fixate on a moving object, they can follow it with a continuous tracking movement called smooth pursuit. This type of eye movement cannot be evoked voluntarily, but only occurs in the presence of a moving object. Periods of fixation typically end after some hundreds of milliseconds, after which a new saccade will occur (Goldberg, 2000). The eyes normally move in lock-step, making equal, conjunctive movements. For a close object, the eyes need to turn towards each other somewhat to correctly image the object on the foveae of the two eyes. These disjunctive movements are called

vergence, and rely on depth perception (see Figure 8.2). Since the eyes are located on the head, they need to compensate for any head movements that occur during fixation. The vestibulo-ocular reflex uses inertial feedback from the vestibular system to keep the orientation of the eyes stable as the eyes move. This is a very fast response, but is prone to the accumulation of error over time. The opto-kinetic response is a slower compensation mechanism that uses a measure of the visual slip of the image across the retina to correct for drift. These two mechanisms work together to give humans stable gaze as the head moves.

The ocular-motor system implemented on Cog and Kismet is an approximation of the human system. Kismet's eyes periodically saccade to new targets chosen by the targeting system, tracking them smoothly if they move. On Kismet, vergence eye movements are not made, since with the scale of the head the robot ends up looking disturbingly cross-eyed even if it is correctly verged. On Cog, a disparity measure computed by comparing left and right camera views drives vergence movements which simply add to whatever tracking movements are being made – the conjunctive and disjunctive channels operate independently. An analogue of the vestibular-ocular reflex has been developed for Cog using a 3-axis inertial sensor. A crude approximation of the opto-kinetic reflex is performed by our implementation of smooth pursuit.

8.3 Social gaze

Apart from functional constraints, eye movements in humans also have communicative value. This needs to be considered when designing humanoid robots. To a human observer, the robot's eyes indicate its locus of attention. The robot's degree of engagement can also be conveyed, to communicate how strongly the robot's behavior is organized around what it is currently looking at. If the robot's eyes flick about from place to place without resting, that indicates a low level of engagement, appropriate to a visual search behavior. Prolonged fixation with smooth pursuit and orientation of the head towards the target conveys a much greater level of engagement, suggesting that the robot's behavior is very strongly organized about the locus of attention. Kismet makes a great deal of use of gaze and head/neck posture to convey state, the goal being to give a cooperative human natural cues as to how they could help the robot. For example, Kismet has a number of coordinated motor actions designed to deal with various limitations of Kismet's visual perception (see Figure 8.3). If a person is visible, but is too distant for their face to be imaged at adequate resolution, Kismet engages in a calling behavior to summon the person closer. People who come too close to

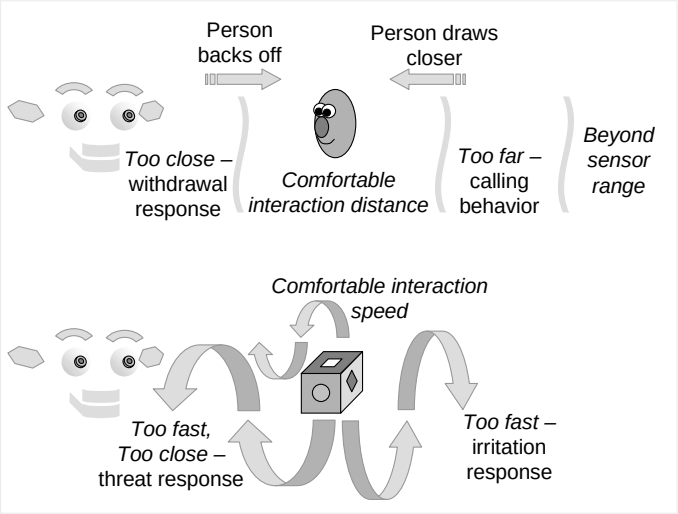


Figure 8.3: Regulating interaction. People too distant to be seen clearly are called closer; if they come too close, the robot signals discomfort and withdraws. The withdrawal moves the robot back somewhat physically, but is more effective in signaling to the human to back off. Toys or people that move too rapidly cause irritation.

the robot also cause difficulties for the cameras with narrow fields of view, since only a small part of a face may be visible. In this circumstance, a withdrawal response is invoked, where Kismet draws back physically from the person. This behavior, by itself, aids the cameras somewhat by increasing the distance between Kismet and the human. But the behavior can have a secondary and greater effect through "social amplification" – for a human close to Kismet, a withdrawal response is a strong social cue to back away, since it is analogous to the human response to invasions of personal space. Similar kinds of behavior can be used to support the visual perception of objects. If an object is too close, Kismet can lean away from it; if it is too far away, Kismet can crane its neck towards it. Again, in a social context, such actions have power beyond their immediate physical consequences. A human, reading intent into the robot's actions, may amplify those actions. For example, neck-craning towards a toy may be interpreted as interest in that toy, resulting in the human bringing the toy closer to the robot. Another limitation of the visual system is how quickly it can track moving objects. If objects or people move at excessive speeds, Kismet has difficulty tracking them continuously. To bias people away from excessively boisterous behavior in their own move-

ments or in the movement of objects they manipulate, Kismet shows irritation when its tracker is at the limits of its ability. These limits are either physical (the maximum rate at which the eyes and neck move), or computational (the maximum displacement per frame from the cameras over which a target is searched for). Here we see the visual-motor system being driven by the requirements of a nominally unrelated sensory modality, just as behaviors that seem completely orthogonal to vision (such as ear-wiggling during the call behavior to attract a person's attention) are nevertheless recruited for the purposes of regulation. These mechanisms also help protect the robot. Objects that suddenly appear close to the robot trigger a looming reflex, causing the robot to quickly withdraw and appear startled. If the event is repeated, the response quickly habituates and the robot simply appears annoyed, since its best strategy for ending these repetitions is to clearly signal that they are undesirable. Similarly, rapidly moving objects close to the robot are threatening and trigger an escape response. These mechanisms are all designed to elicit natural and intuitive responses from humans, without any special training. But even without these carefully crafted mechanisms, it is often clear to a human when Kismet's perception is failing, and what corrective action would help, because the robot's perception is reflected in behavior in a familiar way. Inferences made based on our human preconceptions are actually likely to work.

In general, motor control for a humanoid robot poses challenges beyond issues of stability and accuracy. Motor actions will be perceived by human observers as semantically rich, regardless of whether the imputed meaning is intended or not. This can be a powerful resource for facilitating natural interactions between robot and human, and places constraints on the robot's physical appearance and movement. It allows the robot to be readable – to make its behavioral intent and motivational state transparent at an intuitive level to those it interacts with. It allows the robot to regulate its interactions to suit its perceptual and motor capabilities, again in an intuitive way with which humans naturally cooperate. And it gives the robot leverage over the world that extends far beyond its physical competence, through social amplification of its perceived intent.

8.4 Visual attention

Cog and Kismet have attention systems that are designed to attract the robot towards features that humans find visually salient (Nothdurft, 1993), such as motion, the presence of skin tone, bright colors, and size. The advantage of doing so is that in interactive situations people may intuitively provide the right cues to direct the robot's attention, such as shaking an object, moving it

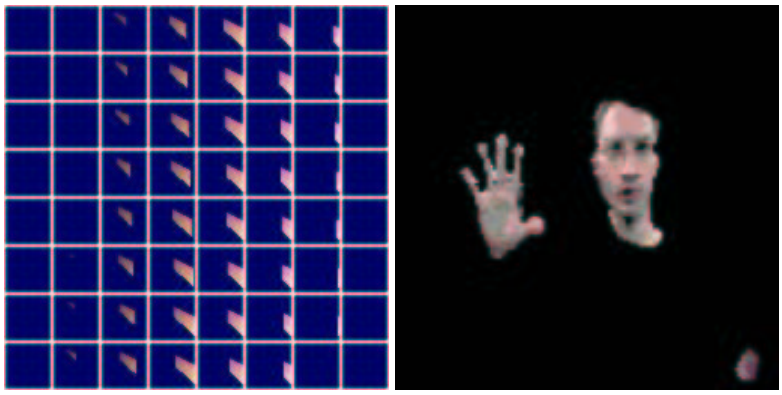


Figure 8.4: The skin tone filter responds to 4.7% of possible (R, G, B) values. Each grid in the figure to the left shows the response of the filter to all values of red and green for a fixed value of blue. The image to the right shows the filter in operation. Typical indoor objects that may also be consistent with skin tone include wooden doors, cream walls, etc.

closer, waving their hand, and so on (see Figures 8.5, 8.6 and 8.7). The attention system can also take care of tasks that need very fast response times. For example, on Kismet looming objects are detected pre-attentively using a measure of optic flow expansion, to facilitate a fast reflexive withdrawal. The output of low-level feature detectors for color and motion are combined through a weighted average to produce a single attention map. This combination allows the robot to select regions that are visually salient and to direct its computational and behavioral resources towards those regions.

The operation of a representative low level feature detector is shown in Figure 8.4). This filter is a simple skin tone detector, a computationally inexpensive means to find regions which may contain faces or hands. Such filters are of course very imperfect, but if no other information is available they are better than simply staring at the wall or ceiling for hours on end (which seems to be what all robots do be default). Higher level modules can compensate for their limitations when the opportunity arises – for example, both Cog and Kismet use the frontal face detector developed by Viola and Jones (2001).

Figure 8.5: These images are from a sequence in which the instructor wanted the robot to attend to the green object as it moved away from a central location. In the first image the robot is clearly attending; in the second it just as clearly has become fixated on the instructors face. Knowing this prompted the instructor to wave the object a little until it regained the robot's attention.

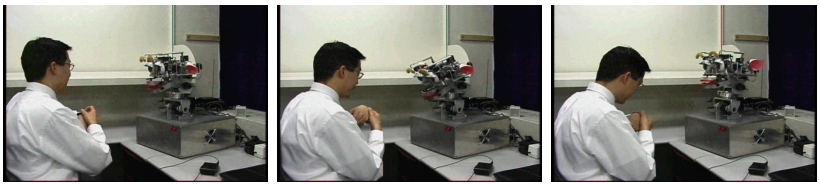


Figure 8.6: Kismet interacting with a test subject unacquainted with the details of the robot's implementation and behavior. The subject decides to show Kismet his watch, so he brings it forward and taps it (left). Motion plus size plus skin-color makes this an attractive target and Kismet looks at it (center). Once the motion stops, Kismet looks away (right).

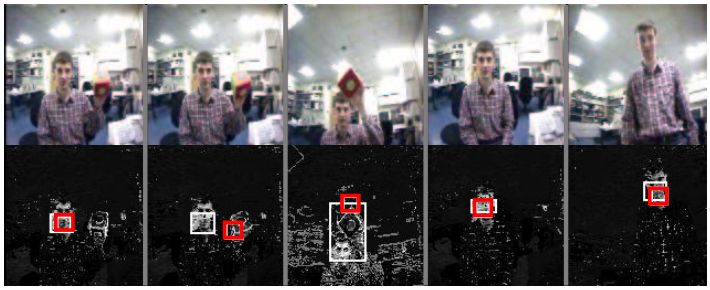


Figure 8.7: Manipulating low-level attention. Images on the top row come directly from the robot's camera. Images on the bottom summarize the contemporaneous state of the robot's attention system. Brightness in the lower image corresponds to salience; rectangles correspond to regions of interest. The thickest rectangles correspond to the robot's locus of attention. In the first pair of images, the robot (Kismet) is attending to a face and engaging in mutual regard. By shaking the colored block, its salience increases enough to cause a switch in the robot's attention. The third pair of images shows that the head tracks the toy as it moves, giving feedback to the human as to the robot's locus of attention. The eyes are also continually tracking the target more tightly than the neck does. In the fourth pair of images, the robot's attention switches back to the human's face, which is tracked as it moves.

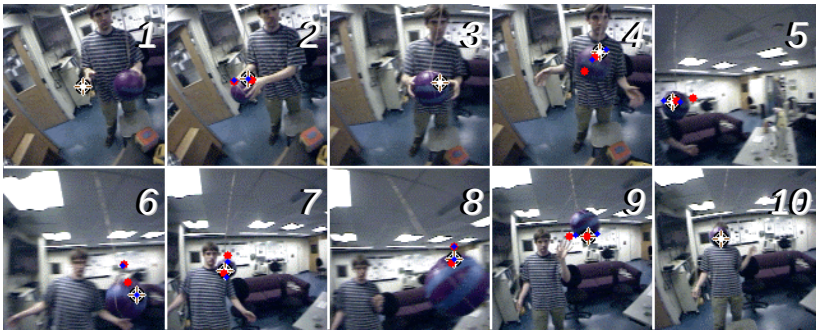


Figure 8.8: The tracking system has an inbuilt preference for moving objects. This mean that if the robot is staring at something, and you want to bring the robot's attention to it, then you can simply move the object in front of the robot (frames 1 to 2). This figure shows frames from a 9 second sequence of the robot (Cog) tracking a rapidly moving ball suspended from the ceiling by elastic until it crashes into the author's face. The crosshair shows the current tracked target. Notice the wide range of motion (compare frames 1, 5, 10) and distance to the ball (compare frames 7 and 8) over this short sequence. The tracker continually looks several pixels forward and backward along its direction of motion and then shifts to whichever location is moving most rapidly. This keeps the tracker on the target, although it will not stay locked to a specific part of the target.

8.5 Maintaining gaze

In the presence of multiple salient objects, it is useful be able to commit attention to one of the objects for a period of time. This gives time for post-attentive processing to be carried out on the object, and for downstream processes to organize themselves around the object. On Kismet, an example of this is visual search – the robot scans across the visual field, dwelling long enough at each point of fixation to decide whether the fixated object is behaviorally relevant (for example, it may lack eyes, which are searched for post-attentively). On Cog, the robot needs to keep an object fixated for some time for vergence to stabilize and to get an accurate measure of its distance. Committing to an object is also useful for behaviors that need to be atomically applied to a target. On Kismet, an example is a calling behavior where the robot needs to stay looking at the person it is calling for the gesture to be socially readable. On Cog, while poking an object the robot needs to maintain fixation.

Figure 8.9: Behavior of the tracker over a longer period (on Kismet). Frames are taken at one second intervals. The white squares indicates the position of the target. The target is not centered in the images since they were taken from a camera fixed with respect to the head. On the third row, the face slips away from the tracker, but it is immediately reacquired through the attention system. The images are taken from a three minute session during which the tracker slipped five times. This is typical performance for faces, which tend not to move too rapidly.

To allow such commitment, the attention system is augmented with a tracker. The tracker follows a target in the visual field, using simple correlation between successive frames. Usually changes in the tracker target will be reflected in movements of the robot's eyes, unless this is behaviorally inappropriate. If the tracker loses the target, it has a good chance of being able to reacquire it from the attention system. Figures 8.8 and 8.9 shows the tracker in operation on Cog and Kismet respectively.

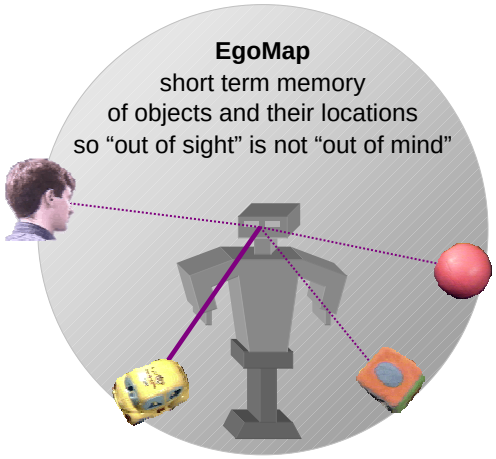


Figure 8.10: The egocentric map or 'egomap.' Locations of objects are tracked with respect to the robot.

8.6 Maintaining an egocentric map

Low-level attention and visual tracking have no memory for objects – once something goes out of view, it is completely forgotten. This does not make for a very useful robot. To rectify this deficit, a map of objects the robot has observed recently is maintained. The map is egocentric; locations are expressed relative to the robot's own position. In fact, the distance to objects is ignored and only the direction to them is stored. The purpose of the map is to allow the robot to return its gaze to an object it has previously observed (even if that object has left the field of view), and this does not require knowledge of distance unless an object is extremely close. A kinematic model of the head is used to computed eye gaze relative to the robot's torso when it is fixating an object. If the object is familiar to the robot from poking, its identity is stored in a two-dimensional grid of bins indexed by spatial dimen-

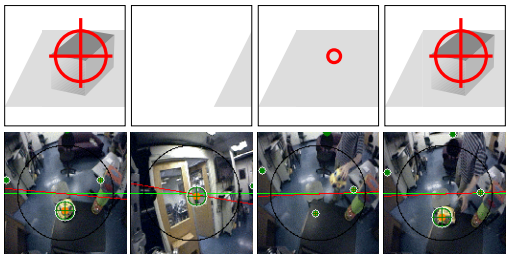


Figure 8.11: Keeping track of locations. Circles with cross-hairs represent locations that contain a particular object. If the object is removed, this is detected using color histograms (Swain and Ballard, 1991), and is indicated by a small circle without a cross-hair. The upper row is a cartoon sequence to illustrate what is happening in the views below, which are taken directly from Cog's egocentric map. Initially a yellow car is present on the table in front of Cog. The robot looks away to the door, and when it looks back, the car is no longer present. It then reappears and is immediately detected. This behavior, along with object tracking (which has also been implemented), give the basics of a representation of the robot's workspace.

sions similar to latitude and longitude (see Figure 8.10), along with the time of observation. Later, if the robot needs to refixate an object, it consults the grid for the last observation of the object, and then directs the robot's gaze to turn to approximately the right direction. Once there, the object recognition module is requested to locate the object in question, and the robot will fixate it precisely.

8.7 Flat-track compound coordinate system

Another important aspect of objects is their pose. This section develops a pose tracking mechanism designed for objects that can be recognized in one pose but not more generally. This is an important scenario for this thesis since poking only offers segmented views of the part of an object facing the camera. Head tracking will initially be treated. It is a useful case in and of itself, and is suitable because face detection is currently better developed for frontal presentations than arbitrary pose. Another reason to work with head tracking is that there are data sets available for evaluating the fidelity head pose estimation.

In 3D space, there are six dimensions associated with the pose of a rigid

object – three translational, and three rotational. When tracking objects using a camera, changes in some of these pose dimensions can be difficult to recover accurately. One approach to deal with this is to have a strong model of the object being tracked, which has been particularly successful for head tracking (see for example Black and Yacoob (1995)). The shape of the human head is broadly similar across the species. Anthropometry characterizes the distribution of face length scales and ratios within different sub-groups. These distributions are quite narrow for a subject whose gender, race, and age are known. Horprasert et al. (1997) make use of this to estimate head orientation from monocular images. They show that pose can be recovered by tracking just five points on the face (four at the eye corners and a fifth at the tip of the nose), given that the necessary anthropometric data is available. They propose a two stage system that estimates a subjects gender, race and age first, indexes into the appropriate table of anthropometric data, and then performs the pose estimation. At the other end of the scale, there are pose tracking systems which do not require a prior model, and are therefore of more general application than systems that rely on special characteristics of the head – for example Harville et al (Harville et al., 1999). Other points on the spectrum include the application of eigenspace techniques to directly recognize the pose of a specific user, as opposed to tracking changes in pose (McKenna and Gong, 1998). And then there are very many systems designed to run in real-time, using a wide variety of simple cues such as hair outline (Wang and Brandstein, 1998).

Some representations are more prone to accumulated errors than others, and there has been considerable research on good coordinate systems for tracking under various situations. Yet another one is introduced here, with the goal being to minimize the effect of mis-estimation of object shape on tracking, and to allow for opportunistic calibration whenever a known view of the object is observed. If a rigid object being viewed by a camera does not rotate in depth but is otherwise free to explore a 3D space, the object's pose can be specified completely with just four numbers :-

- ▷ A position on the image plane, specified by two coordinates, giving a ray from the camera to a particular point on the object.
- ▷ A coordinate specifying any rotation of the object in the plane parallel to the camera, which is the only rotational freedom the object has, given that it cannot (for now) rotate in depth.
- ▷ A coordinate specifying a scaling of the projection of the object on the image plane, or any other scalar such as size or metric distance to the object.

These coordinates completely describe the object's pose in the sense that if the camera configuration is known, and *if the shape of the object is known*,

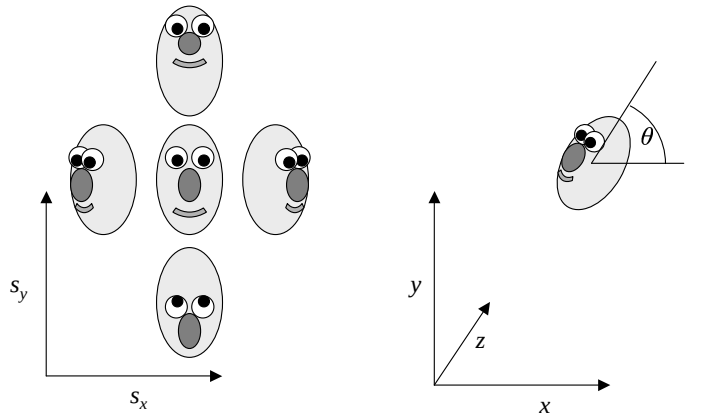


Figure 8.12: The 'flat-track' compound coordinate system. One component is a surface coordinate on the object being tracked (left). The second component are the degrees of freedom of a flat surface facing the camera – x and y position, a measure of scale or distance, and an in-plane rotation.

the full 3D pose of the object can be recovered from these parameters. The need for the shape of the object arises from the implicit reference points of the coordinates.

Once the object starts rotating in depth, there are two more degrees of freedom to factor in. The goal here to introduce them without destroying the simplicity of the image plane coordinates defined above. Suppose the object being tracked is basically convex (if it has a more awkward shape, there is a good chance that pose can be recognized from its silhouette directly). Then at any moment there will ideally be a unique region on the surface of the object that is close to parallel to the image plane. As the object rotates in depth, this region will shift to another part of the surface. We can parameterize where this region lies on the surface of the object using two dimensions. And since the region is (by construction) parallel to the image plane, the four coordinates developed earlier can be recast as follows :-

- ▷ Two coordinates that specify where the projection of the parallel region lies on the image plane.

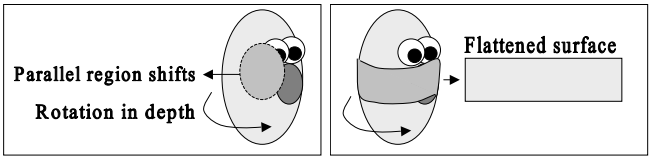


Figure 8.13: Left: when an object rotates in depth, a different region of the object will become parallel to the image plane. Right: if the regions of the object that become parallel during a movement of the head do not explore the surface in 2D, then the surface they do explore may be thought of as Euclidean without running into serious contradictions (except for full 360° excursions).

- ▷ A coordinate specifying how the parallel region is rotated with respect to the image plane. This is the only rotational degree of freedom the parallel region has, by construction.
- ▷ A coordinate specifying a scaling of the parallel region (or equivalently of the projection of the entire object, as before).

Combined with two coordinates that determine what part of the surface of the object is currently parallel to the image plane, we have a 6-dimensional coordinate system that fully specifies the 3D pose of the object (if the shape of the object is known). This choice of coordinates has some virtues. In contrast to Euler angles, for example, the coordinates can be considered separately and in any order. This is least obvious for the rotation coordinate, but becomes clear if that coordinate is thought of as a counter-rotation of the camera about its optical axis.

A crucial issue that has not yet been addressed is what kind of coordinates are used to span the surface of the object being tracked. There are many possible coordinate systems for specifying a location on a convex surface – for example, latitude and longitude angles. The challenge here is to use coordinates that can be related to the projection of the object without knowledge of its 3D shape. There is no such magical coordinate system, so technically at this point the dimensions of the objects have to be estimated before proceeding any further. But suspending disbelief for a moment, consider setting up a Euclidean coordinate system on the surface (which can be thought of as flattening the surface out onto a plane and then using standard rectangular coordinates). Of course, it isn't possible to flatten out the surface in this way without introducing inconsistencies. But if we do so anyway, then coordinates on the surface of the object that lie within the parallel region will map on to the image plane very simply, with just a scaling and an in-plane rotation. If we only ever try to relate coordinates within this region, then we can relate small steps in the image plane to small steps on the surface of the object, and

so integrate the surface coordinates without needing to know the actual shape of the object.

The above discussion imposes two conditions :-

1. We must be able to determine what part of the projection of the object originated from a surface parallel to the image plane.

2. The path the parallel region traces across the surface of the object must lie within a strip that is thin relative to the curvature of the object. The wider the strip, the less Euclidean it is. The strip also must not make a full 360° excursion, no matter how thin it is.

The first condition is tractable and will be addressed shortly. With regard to the second condition: in practice, the estimated curvature of the object should be factored in to the surface coordinate system, and this becomes an argument about what kinds of movements the accuracy of the estimate actually matters for. The answer is as might be expected: tracking accuracy is insensitive to the estimate of shape for movements combining in-plane translation, scaling (translation in depth), in-plane rotation, and rotation in depth for which all the surface patches made successively parallel to the image plane lie within a strip. This includes the important case of turning away, then turning back in an approximately symmetric manner.

8.7.1 Pose detector

The human face has a rich enough structure to admit of several possibilities for pose recognition :-

- ▷ Frontal pose. Because of the approximate bilateral symmetry of the human body, the projection of the face is close to symmetric in this pose. This, along with the relatively clear-cut features of the face such as the eyes and nose, makes the pose relatively easy to detect. This pose also has special behavioral status because of attentive orienting, and occurs very frequently during face-to-face human/robot interaction, which is my domain of interest.
- ▷ Profile view. The head has its most well-defined silhouette for 90° of yaw, particularly around the nose.
- ▷ Hair-line. Wang et al (Wang and Brandstein, 1998) argue that the hair-line can be indicative of pose.
- ▷ Recognition of trajectories. This is a rather different possibility, where the output of relative tracking is used as a feature in its own right. The movement of the head is strongly constrained by the neck, and it seems

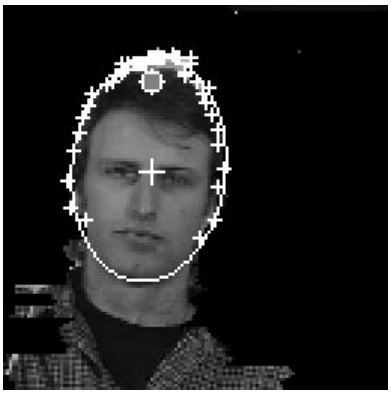


Figure 8.14: Finding the outline of the head once it has been detected. Probes (shown as small crosses) radiate out from a location near the top of the head/body (shown as a circle). Across a large range of scales, the silhouette of the head can be extracted from the contour points encountered by the probes, and matched against an elliptic model.

possible that those constraints may be tight enough to give unambiguous interpretations for certain types of head movement, particularly if enriched with some knowledge of the head outline.

Frontal pose is the option adopted here. Profile was problematic for extracting an accurate orientation, since the silhouette changes too slowly with changes in roll and yaw. The hair-line is very variable between subjects. And trajectory recognition would in practice require a great deal of training data to learn the priors, and even then it is not clear whether it would actually do anything.

The outline of the head is tracked using a collection of techniques that seem to be typical of real-time systems (Cordea et al., 2000), such as image differencing and ellipse-fitting. The implementation described here is qualitatively similar to that of Smith-Mickleson (Smith-Mickelson, 2000), and also traces back to Birchfield's work (Birchfield, 1998).

Before the head outline can be tracked, the head needs to be detected in the first place. Head movements often have a marked translational component. This is particularly the case when someone is walking into the scene (which is a perfect time to do initialization). Such movement makes it relatively easy to distinguish the head and body from the background using image differencing. A simple template tracker is assigned to the largest blob detected in this way. The image is then modeled as being generated by overlaying two layers, a "body layer" moving with the tracker, and a "background

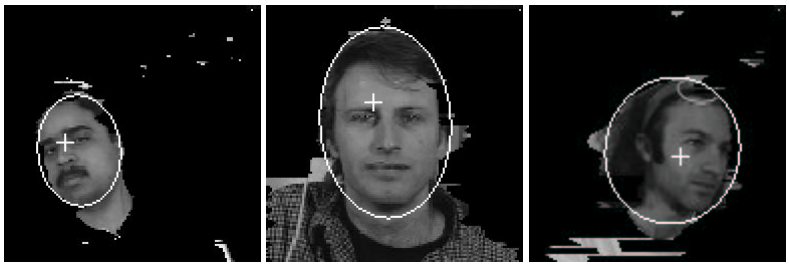


Figure 8.15: Snapshots of the head tracker in action. Hair is problematic. Sometimes it will be included, sometimes not. In individuals with a great deal of hair (rightmost figure), the ellipse can deviate a great deal from the basic shape of the head itself.

layer" that is stationary. Each pixel is independently assigned to one of the layers based on the intensity difference of that pixel with its predicted location in the previous frame for each layer. This gives a cleaner, more persistent outline for the body than raw image differencing, and discounts at least some fraction of pixels from moving objects in the background. The outline is cleaned up using various heuristics (implemented using Viterbi-based optimization across scan-lines). Probes are sent out in all directions from a point close to the top of the body to characterize the outline, and in particular identify the location of the head. The probes are filtered to eliminate those that wander back to the body, and an oriented ellipse is fit to the remainder (Pilu et al., 1996). Figure 8.14 shows an example of this.

If the ellipse is a good fit, its interior is used to initialize a color histogram. There is a tradeoff between trying to include the hair-color within the histogram while avoiding including the background. I found it necessary to err on the side of caution, and not include the contents of the entire ellipse in the histogram, which often meant that hair color was not included. Figure 8.15 shows examples of the kind of variation this can lead to in what the putative "head outline" actually means.

At the low resolutions real-time performance mandates, many researchers attempt to locate eyes using the fact that since they are generally somewhat recessed, their associated pixels tend to be darker than the surrounding skin. But since the face may be unevenly illuminated, it is difficult to translate this model into a statement about pixel intensity or color thresholds (for example). A statement about intensity or color *differences* seems more tractable (Sinha, 1994) but as a differential measure this is subject to noise for small regions such as the eyes.

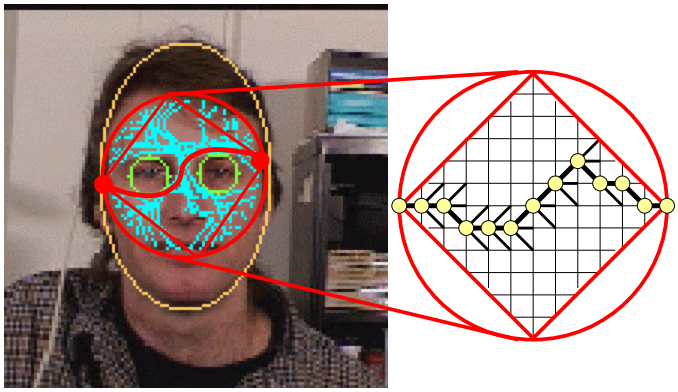


Figure 8.16: How a path is found between two example end-points. A grid is laid down within an area between the end-points as shown. For the orientation shown in the figure, a path starts at the left, and moves through successive grid intersections as shown, always moving right and moving at most one step up or down (giving a maximum slope of 45°). By analogy with HMMs, each step to the right is a time step, each possible vertical level is a state, and the transition matrix is sparse with three entries per state. With the simple cost function described in the text, the optimal path can be efficiently computed with a Viterbi lattice structure. The grid resolution shown is artificially low for clarity. The path picked out on the left might just as easily have gone under or over both eyes, that is not relevant to localization.

The approach adopted here is to use a relative measure whose support extended across a large fraction of the face. Paths between different points on the face are considered, where each path is assigned a cost that sums the distance of its individual pixels from a simple model of skin color (and since as mentioned above eyes are often recessed and relatively poorly illuminated from overhead light, intensity is also factored in). Paths which pass through an eye will have a higher cost than paths that detour around the eye. A Viterbi-based calculation is used to assign optimal paths to pairs of end-points on opposite sides of the face, searching over all paths that remain within the face and don't exceed some maximum curvature (see Figure 8.16). Each of these paths is computed from about one half of the pixels on the face. The paths are then combined to localize the eyes, which correspond to regions avoided by the paths. A series of heuristic tests based on the paths around avoided regions serve to distinguish actual eyes from regions that are simply not quite as attractive as the neighboring area.

Figure 8.17: Frontal pose being recognized for a number of individuals. As noted in the text, roll of the head is not problematic since it will be parallel to the image plane and so can be recovered directly from the angle the eyes make with the horizontal.

Only pairs of avoided regions roughly aligned horizontally are considered. The regions give a reasonable estimate of the deviation from the horizontal of the angle between the eyes, which will be useful for initializing the roll. The location of the bridge of the nose serves as a useful origin for the surface of the face. The degree of symmetry can be estimated and used to see if the pose is close enough to zero yaw for initialization to be practical. Pitch is initialized by comparing the bridge location with the head outline as determined by the head tracker. The estimated size of the eyes, the distance between them, and the dimensions of the head outline all can contribute to an estimate of the size of the head (code for: none of them are at all reliable). The size estimate is only a relative measure across the interaction, since there is a scale factor that can't be recovered.

To actually implement a pose tracking system based on this coordinate system, a mesh is laid down on the projection of the head as illustrated in Figure 8.18. Nodes on the mesh are kept in correspondence to the face using simple template trackers, which are destroyed if they misbehave (measured by a set of consistency checks) and recreated elsewhere. Scaling, in-plane rotation, and in-plane translation are straightforward to compute from deformations of this mesh. As the head rotates in depth, some trackers will lose

Figure 8.18: Mesh initialization. The mesh is initially distributed arbitrarily. It is pruned by the head outline when it is detected, and by heuristics based on the relative motion of different parts of the mesh. When frontal pose is detected, surface coordinates of the mesh can be initialized within the parallel region (that part of the face that is parallel to the image plane).

the support of the surface they are tracking as it becomes occluded, and so be destroyed. New parts of the head will become visible and have trackers assigned to their surface.

The mesh is used to maintain the surface coordinate system as follows. First, the parallel region is determined heuristically. If the translational component of motion can be eliminated, the parallel region can be identified easily because the flow due to rotation peaks there (since the motion of that surface is completely parallel parallel to the image plane). Translational motion can be accounted for by normalizing flow relative to the outline of the head. This crude procedure works better than it should because in practice translations and rotations of the head are often coupled so as to sum within the parallel region rather than cancel. Exceptions include pure rolls and translations in depth. The extent of the parallel region is chosen to scale in a heuristic way with the head outline, since in theory it should be infinitesimally small but in practice it has to be assigned some extent to be useful. And luckily, surface distortions such as the nose don't seem to cause trouble.

The parallel region can be seen as a mask overlaid on the image, within which it is safe to relate image coordinates and surface coordinates. Pose recognition events, detected in the manner described in the previous section, are used to choose an origin on the surface, and an initial translation, scaling and (in-plane) rotation of the surface coordinate system with respect to the image plane. This association is represented by assigning surface coordinates to points on the mesh that lie within the parallel region, augmenting the image plane coordinates they jointly possess. As the parallel region shifts during a rotation in depth, new points entering the region are assigned surface coordinates based on their image plane coordinates, with the transformation

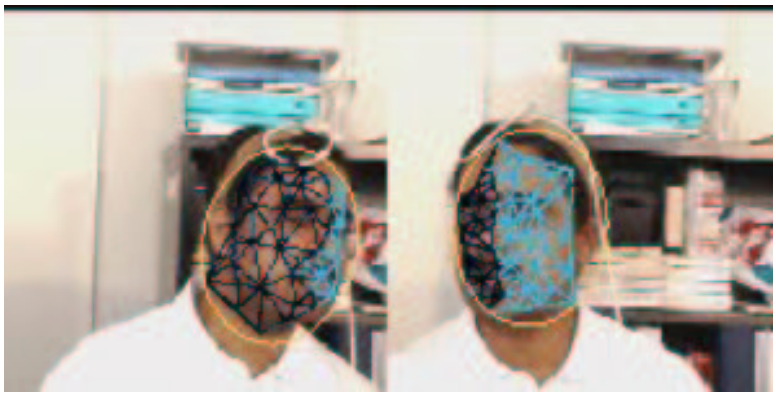


Figure 8.19: A visualization of surface coordinates on the mesh. The mesh has been colored here based on the sign of a surface coordinate, so that it appears as two halves locked onto either side of the face.

between the two easy to maintain using the rotation, scaling, and translation of the mesh already recovered.

Independently of the argument given earlier for the types of movements that can be tracked without accurate knowledge of the shape of the head, the mesh allows a new set of trajectories to be tracked: those which leave some portion of the face visible throughout. The surface coordinates of points on the mesh covering that part of the face can be used as landmarks.

Recovery of the 3D location of the head is straightforward, given knowledge of the camera's parameters, although there is of course a scale/depth ambiguity since no absolute depth information is recovered. Recovery of 3D orientation is equally straightforward, but shape dependent. The output of the tracker is effectively a procedure for turning a specified point on the surface of the object towards the camera and then rotating it to a specified degree. To convert this into Euler angles, for example, requires knowledge of the shape of the object so that surface points can be associated with vectors from wherever the center of the head is taken to be. At this point, we must make use of the estimates for the dimensions of the head from the head tracker and make the conversion using a simple ellipsoidal model. The crucial point is that inaccuracies in this process do not feed back to the tracker itself.

8.7.2 An evaluation

The system was tested on a data-set made available by Sclaroff et al (La Cascia et al., 2000), consisting of video of head movements with ground truth

measured by a Flock of Birds sensor on the subjects' heads. These sequences are 200 frames in duration. To test the stability of the tracker over long intervals, the Sclaroff sequences are here artificially extending by looped them forward and back for twenty iterations. Figure 8.20 shows tracking results for the sequence which appeared to have the largest rotation in depth (in no case unfortunately did the eyes become occluded, which would have made for a better demonstration of the advantages of the system developed in this paper). Angular measurements are limited by the accuracy with which they can be initialized, which turns out to be to within about 5° for roll and yaw, and about 10° for pitch. Because of re-initialization events, estimates of pose will contain discontinuities when drift is corrected, which is not brought out in the figure. This could be dealt with for estimation of pose across a pre-recorded video sequence like this one, but for use in a vision interface it seems the discontinuities are unavoidable. This is because the best estimate of the current pose does truly change instantaneously when an initialization even occurs, and there is no point propagating information backwards to previous frames during real-time interaction unless there is some background processing going on that can have high latency.

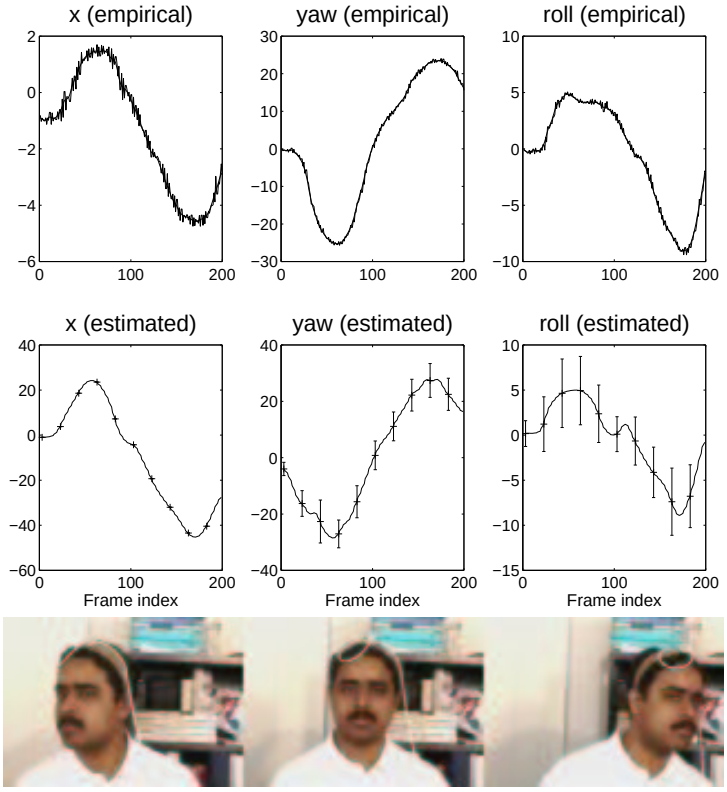


Figure 8.20: Results for a sequence containing a yaw movement and horizontal translation, with all other parameters remaining basically unchanged except for a slight roll. The top row shows ground truth. The second row shows the estimated pose parameters that change significantly during the sequence. The estimated x coordinate is left in terms of the image plane. Values plotted are averaged for each occurrence of a particular frame over a *single tracking run* constructed from a sequence being played, then played in reverse, then repeated again for twenty iterations. Error bars show the standard deviation of estimates for each frame. There is about a 5° error in angles, which in this case means the roll estimate is mostly noise.

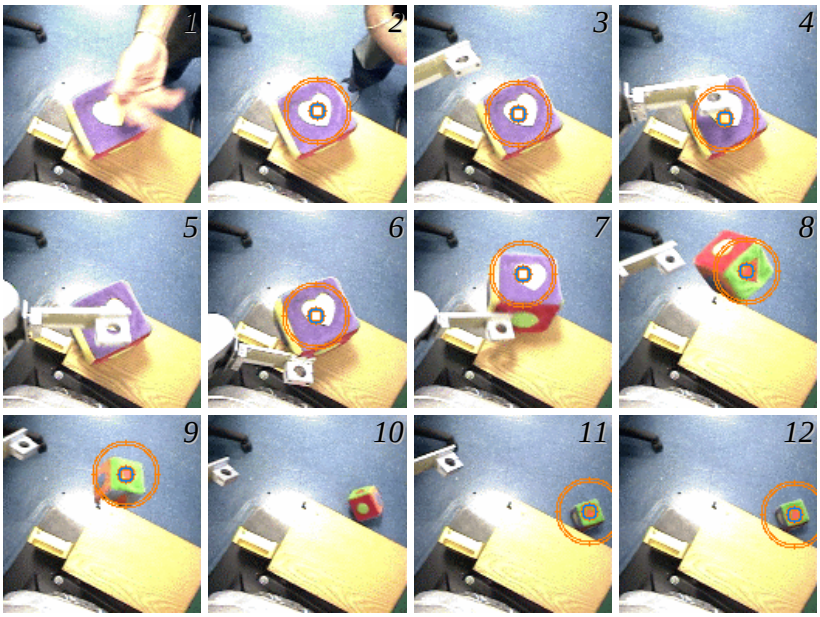


Figure 8.21: Tracking a poked object (a cube). The robot is familiar with the purple and green face of the cube. By tracking the cube using flat-track,the two faces are revealed to be views of the same object.

CHAPTER 9

First words: working with speech

> *The trouble with having an open mind, of course, is that people will insist on coming along and trying to put things in it.* (Pratchett, 1989)

Speech sounds form a special category of percept, since speech is very much a cultural invention. Many of its properties are simply agreed to, rather than grounded in any immediate physical necessity There are of course many physical constraints on speech, but within that space there is huge potential diversity. And in fact, as a communication protocol, speech is very flexible. There are special schemes for talking to children, or pets. So it is quite easy to imagine that we could borrow one of these schemes for robots. One of the goals of the Kismet robot in our group was to evoke the "motherese" style of speech, for functional benefits (Varchavskaia et al., 2001). There are many robotics projects looking at various aspect of speech such as the development of vocabulary and/or grammar from various forms of experience (Roy and Pentland, 2002; Steels, 1996). The goal of this chapter is to produce a real-time system for extending vocabulary, augmented with a slower offline process for refinement, just as was the case for object recognition in Chapter 5.

9.1 The microphones

A natural-language interface is a desirable component of a humanoid robot. In the ideal, it allows for natural hands-free communication with the robot

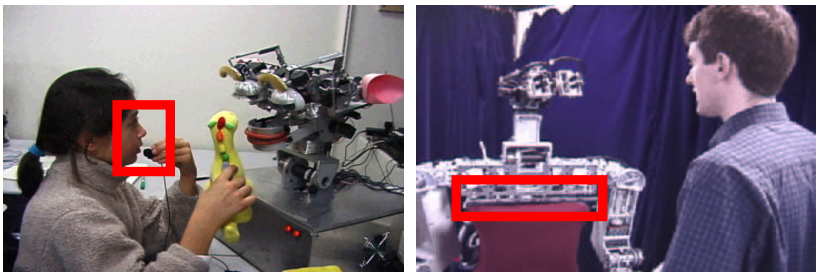


Figure 9.1: For Kismet, a clip-on/hand-held microphone was used (left). Anyone interacting with the robot needed to be informed of this. Experience showed that people would frequently forget to use the microphone if it was not clipped on – it was not an intuitive interface for face-to-face communication. On Cog, a commercial microphone array was installed (right), with a clip-on microphone available for when there was excessive background noise – Cog is located right beside a busy machine shop. This meant that the robot could always respond to voice in its vicinity, and if its responses were poor, the human could either move closer, speak louder (natural responses) or pick up the back-up microphone. The transition between microphones is handled automatically.

without necessitating any special skills on the human user's part. In practice, we must trade off flexibility of the interface with its robustness. The first trade-off is the physical interface. For best results with contemporary speech recognition techniques, a high-quality microphone close to the mouth is desirable. On Kismet, a wireless clip-on or hand-held microphone was used, as shown in Figure 9.1. This caused some difficulties, because given Kismet's anthropomorphic face and prominent bright-pink ears, people expected the robot to be able to hear them directly without any intermediary. Unfortunately placing microphones in the ears or anywhere else in the head would have been completely useless, since all the motors controlling facial features were very noisy (and the ear motors were perhaps noisiest of all). On Cog, a microphone array was installed across its torso. This meant that a person interacting with the robot did not need to be instrumented, and natural human behavior when they want to be heard – speaking louder or coming closer – did in fact make the robot hear better. If background noise is high, there is an auxiliary wireless microphone which subsumes the microphone array.

9.2 Infant-directed speech

A crucial factor for the suitability of current speech recognition technology to a domain is the expected perplexity of sentences drawn from that domain. Perplexity is a measure of the average branching factor within the space of possible word sequences, and so generally grows with the size of the vocabulary. For example, the basic vocabulary used for most weather-related queries may be quite small, whereas for dictation it may be much larger and with a much less constrained grammar. In the first case speech recognition can be applied successfully for a large user population across noisy telephone lines (Zue et al., 2000), whereas in the second a good quality headset and extensive user training are required in practice. It is important to determine where robot-directed speech lies in this spectrum. This will presumably depend on the nature of the task to which the robot is being applied, and the character of the robot itself. We evaluated this for Kismet (Varchavskaia et al., 2001). When interacting with a youthful-appearing robot such as Kismet, our hope was that the speech input may have specialized characteristics similar to those of infant-directed speech (IDS). In particular, we were interested in the following:

- ▷ Does speech directed at Kismet include a substantial proportion of single-word utterances? Presenting words in isolation side-steps the problematic issue of word segmentation.
- ▷ How often, if at all, is the speech clearly enunciated and slowed down compared to normal speech? Overarticulated speech may be helpful to infants, but can be challenging for artificial speech recognizers trained on normal speech.

Whether isolated words in parental speech help infants learn has been a matter of some debate. It has been shown that infant-directed utterances are usually short with longer pauses between words (see for example Werker et al. (1996)), but also that they do not necessarily contain a significant proportion of isolated words (Aslin et al., 1996). Another study (Brent and Siskind, 2001) presents evidence that isolated words are in fact a reliable feature of infant-directed speech, and that infants' early word acquisition may be facilitated by their presence. In particular, the authors find that the frequency of exposure to a word in isolation is a better predictor of whether the word will be learned, than the total frequency of exposure. This suggests that isolated words may be easier for infants to process and learn. Equally importantly for us, however, is the evidence for a substantial presence of isolated words in IDS: 9% found in Brent and Siskind (2001) and 20% reported in Aslin et al. (1996). If Kismet achieves its purpose of eliciting nurturing behavior

from humans, then we would expect a similar proportion of Kismet-directed speech to consist of single-word utterances.

The tendency of humans to slow down and overarticulate their utterances when they meet with misunderstanding has been reported as a problem in the ASR community (Hirschberg et al., 1999). Such enunciated speech degrades considerably the performance of speech recognition systems which were trained on natural speech only. If we find that human caretakers tend to address Kismet with overarticulated speech, its presence becomes an important issue to be addressed by the robot's perceptual system.

A study was made of interactions between young children and the Kismet robot in the context of teaching the robot new words. The sessions were organized by the MIT Initiative on Technology and Self. During these sessions, the robot was engaging in proto-conversational turn-taking, where its responses to utterances of the children were random affective babble. A very minimal mechanism for vocal mimicry and vocabulary extension was present. The purpose of the study was to identify ways to improve the speech interface on the robot based on a better knowledge of the properties of speech directed at this particular robot.

During these experiments the robot was engaging in proto-conversational turn-taking as described in Breazeal (2000), augmented with the following command-and-control style grammar. Sentences that began with phrases such as "say", "can you say", "try" etc. were treated as requests for the robot to repeat the phonetic sequence that followed them. If, after the robot repeated a sequence, a positive phrase such as "yes" or "good robot" was heard, the sequence would be entered in the vocabulary. If not, no action was taken unless the human's next utterance was similar to the first, in which case it was assumed to be a correction and the robot would repeat it. Because of the relatively low accuracy of phoneme-level recognition, such corrections are the rule rather than the exception.

Video of 13 children aged from 5 to 10 years old interacting with the robot was analyzed. Each session lasted approximately 20 minutes. In two of the sessions, two children are playing with the robot at the same time. In the rest of the sessions, only one child is present with the robot. We were interested in determining whether any of the following strategies are present in Kismet-directed speech:

- ▷ single-word utterances (words spoken in isolation)
- ▷ enunciated speech
- ▷ vocal shaping (partial, directed corrections)
- ▷ vocal mimicry of Kismet's babble

A total of 831 utterances were transcribed from the 13 sessions of children playing with the robot. We observed a wide variation of strategies among

subjects. The following preliminary results include a measure of standard deviations, which are mentioned to give an idea of the wide range of the data, and should not be read to imply that the data follows a Gaussian distribution. The total number of utterances varied from subject to subject in the range between 19 and 169, with a mean of 64 (standard deviation of 44, based on a sample of 13) utterances per subject.

Isolated words

These are fairly common; 303 utterances, or 36.5% consisted of a single word said in isolation. The percentage of single-word utterances had a distribution among subjects with a mean at 34.8 and a deviation of 21.1. Even when we exclude both greetings and the robot's name from counts of single-word utterances, we get a distribution centered around 20.3% with a standard deviation of 18.5%. This still accounts for a substantial proportion of all recorded Kismet-directed speech. However, almost half the subjects use less than 10% isolated words, even in this teaching context.

Enunciated speech

Also common is enunciated speech; 27.4% of the transcribed utterances (228) contained enunciated speech. An utterance was counted as "enunciated speech" whenever deliberate pauses between words or syllables within a word, and vowel lengthening were used. The count therefore includes the very frequent examples where a subject would ask the robot to repeat a word, e.g. "Kismet, can you say: GREEN?". In such examples, GREEN would be the only enunciated part of the utterance but the whole question was counted as containing enunciated speech. The mean proportion of enunciated speech is 25.6% with a deviation of 20.4%, which again shows a large variation.

Vocal shaping

In the whole body of data we have discovered only 6 plausible instances (0.7%) of vocal shaping. It may not be an important teaching strategy, or it may not be evoked by a mimicry system that is not responding reliably enough to the teacher.

Vocal mimicry

There were 23 cases of children imitating the babbling sounds that Kismet made, which accounts for 2.8% of the transcribed utterances. However, most children did not use this strategy at all.

9.2.1 Discussion

The interaction sessions were not set up as controlled experiments, and do not necessarily represent spontaneous Kismet-directed speech. In particular, on all occasions but one, at some point during the interaction, children were instructed to make use of the currently implemented command-and-control system to get the robot to repeat words after them. In some cases, once that happened, the subject was so concerned with getting the robot to repeat a word that anything else simply disappeared from the interaction. On three occasions, the subjects were instructed to use the "say" keyword as soon as they sat in front of the robot. When subjects are so clearly focused on a teaching scenario, we can expect the proportion of isolated words, for instance, to be unnaturally high.

Note also that as of now, we have no measure of accuracy of the transcriptions, which were done by hand by one transcriber, from audio that sometimes had poor quality. Given the focus of the analysis, only Kismet-directed speech was noted from each interaction, excluding any conversations that the child may have had with other humans who were present during the session. Deciding which utterances to transcribe was clearly another judgment call that we cannot validate here yet. Finally, since the speech was transcribed by hand, we cannot claim a scientific definition of an utterance (e.g., by pause duration) but must rely on one person's judgement call again.

However, this preliminary analysis shows promise in that we have found many instances of isolated words in Kismet-directed speech, suggesting that Kismet's environment may indeed be scaffolded for word learning. However, fluent speech is still prevalent even in a teaching scenario, and so an unsupervised learning algorithm will be needed to find new words in this case. We have also found that a substantial proportion of speech was enunciated. Counter-intuitively such speech can present problems for the speech recognizer, but at the same time opens new possibilities. For an improved word-learning interface, it may be possible to discriminate between natural and enunciated speech to detect instances of pronunciation teaching (this approach was taken in the ASR community, for example in Hirschberg et al. (1999)). On the other hand, the strategy of vocal shaping was not clearly present in the interactions, and there were few cases of mimicry.

9.3 Automatic language modeling

This section develops a technique to bootstrap from an initial vocabulary (distilled perhaps from isolated word utterances) by building an explicit model of unrecognized parts of utterances. The purpose of this background model is both to improve recognition accuracy on the initial vocabulary and to auto-

matically identify candidates for vocabulary extension. This work draws on research in word spotting and speech recognition. We will bootstrap from a minimal background model, similar to that used in word-spotting, to a much stronger model where many more word or phrase clusters have been "moved to the foreground" and explicitly modeled. This is intended both to boost performance on the original vocabulary by increasing the effectiveness of the language model, and to identify candidates for automatic vocabulary extension.

The remainder of this section shows how a conventional speech recognizer can be convinced to cluster frequently occurring acoustic patterns, without requiring the existence of transcribed data.

9.3.1 Clustering algorithm

A speech recognizer with a phone-based "OOV" (out-of-vocabulary) model is able to recover an approximate phonetic representation for words or word sequences that are not in its vocabulary. If commonly occurring phone sequences can be located, then adding them to the vocabulary will allow the language model to capture their co-occurrence with words in the original vocabulary, potentially boosting recognition performance. This suggests building a "clustering engine" that scans the output of the speech recognizer, correlates OOV phonetic sequences across all the utterances, and updates the vocabulary with any frequent, robust phone sequences it finds. While this is feasible, the kind of judgments the clustering engine needs to make about acoustic similarity and alignment are exactly those at which the speech recognizer is most adept.

The clustering procedure adopted is shown in Figure 9.2. An *n*gram-based language model is initialized uniformly. Unrecognized words are explicitly represented using a phone-based OOV model, described in the next section. The recognizer is then run on a large set of untranscribed data. The phonetic and word level outputs of the recognizer are compared so that occurrences of OOV fragments can be assigned a phonetic transcription. A randomly cropped subset of these are tentatively entered into the vocabulary, without any attempt yet to evaluate their significance (e.g. whether they occur frequently, whether they are similar to existing vocabulary, etc.). The hypotheses made by the recognizer are used to retrain the language model, making sure to give the new additions some probability in the model. Then the recognizer runs using the new language model and the process iterates. The recognizer's output can be used to evaluate the worth of the new "vocabulary" entries. The following sections detail how to eliminate vocabulary items the recognizer finds little use for, and how to detect and resolve competition between similar items.

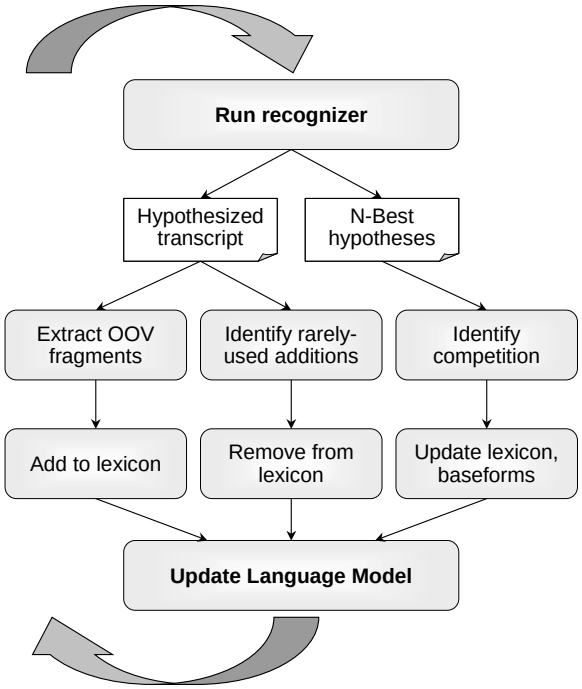


Figure 9.2: The iterative clustering procedure for segmenting speech. A conventional speech recognition system is used to evaluate how useful particular phoneme sequences are for describing the training data. Useful sequences are added to lexicon, otherwise they are dropped.

9.3.2 Extracting OOV phone sequences

The speech recognizer system developed by the Spoken Language Systems group at MIT was used (Glass et al., 1996). The recognizer is augmented with the OOV model developed by Bazzi and Glass (2000). This model can match an arbitrary sequence of phones, and has a phone bigram to capture phonotactic constraints. The OOV model is placed in parallel with the models for the words in the vocabulary. A cost parameter can control how much the OOV model is used at the expense of the in-vocabulary models. This value was fixed at zero throughout the experiments described in this paper, since it was more convenient to control usage at the level of the language model. The bigram used in this project is exactly the one used in (Bazzi and Glass, 2000), with no training for the particular domain.

Phone sequences are translated to phonemes, then inserted as new entries

in the recognizer's lexicon.

9.3.3 Dealing with rarely-used additions

If a phoneme sequence introduced into the vocabulary is actually a common sound sequence in the acoustic data, then the recognizer will pick it up and use it in the next iteration. Otherwise, it just will not appear very often in hypotheses. After each iteration a histogram of phoneme sequence occurrences in the output of the recognizer is generated, and those below a threshold are cut.

9.3.4 Dealing with competing additions

Very often, two or more very similar phoneme sequences will be added to the vocabulary. If the sounds they represent are in fact commonly occurring, both are likely to prosper and be used more or less interchangeably by the recognizer. This is unfortunate for language modeling purposes, since their statistics will not be pooled and so will be less robust. Happily, the output of the recognizer makes such situations very easy to detect. In particular, this kind of confusion can be uncovered through analysis of the N-best utterance hypotheses.

If we imaging aligning a set of N-best hypothesis sentences for a particular utterance, then competition is indicated if two vocabulary items exhibit both of these properties:

- ▷ Horizontally repulsive - if one of the items appears in a single hypothesis, the other will not appear in a nearby location within the same hypothesis
- ▷ Vertically attractive - the items frequently occur in the same location within different hypotheses

Since the utterances in this domain are generally short and simple, it did not prove necessary to rigorously align the hypotheses. Instead, items were considered to be aligned based simply on the vocabulary items preceding and succeeding them. It is important to measure both the attractive and repulsive conditions to distinguish competition from vocabulary items that are simply very likely to occur in close proximity.

Accumulating statistics about the above two properties across all utterances gives a reliable measure of whether two vocabulary items are essentially acoustically equivalent to the recognizer. If they are, they can be merged or pruned so that the statistics maintained by the language model will be well trained. For clear-cut cases, the competing items are merged as alternatives

in the list of pronunciation variants for a single vocabulary unit. or one item is simply deleted, as appropriate.

Here is an example of this process in operation. In this example, “phone” is a keyword present in the initial vocabulary. These are the 10-best hypotheses for the given utterance:

“what is the phone number for victor zue”

```
<oov> phone (nahmber) (mihterz) (yuw)
<oov> phone (nahmber) (mihterz) (zyuw)
<oov> phone (nahmber) (mihterz) (uw)
<oov> phone (nahmber) (mihterz) (zuw)
<oov> phone (ahmberf) (mihterz) (zyuw)
<oov> phone (ahmberf) (mihterz) (yuw)
<oov> (axfaanah) (mberfaxr) (mihterz) (zyuw)
<oov> (axfaanah) (mberfaxr) (mihterz) (yuw)
<oov> phone (ahmberf) (mihterz) (zuw)
<oov> phone (ahmberf) (mihterz) (uw)
```

The “<oov>” symbol corresponds to an out of vocabulary sequence. The sequences within parentheses are uses of items added to the vocabulary in a prior iteration of the algorithm. From this single utterance, we acquire evidence that:

- The entry for (ax f aa n ah) may be competing with the keyword “phone”. If this holds up statistically across all the utterances, the entry will be destroyed.
- (n ah m b er), (m b er f axr) and (ah m b er f) may be competing. They are compared against each other because all of them are followed by the same sequence (m ih t er z) and many of them are preceded by the same word “phone”.
- (y uw), (z y uw), and (uw) may be competing

All of these will be patched up for the next iteration. This use of the N-best utterance hypotheses is reminiscent of their application to computing a measure of recognition confidence in (Hazen and Bazzi, 2001).

9.3.5 Testing for convergence

For any iterative procedure, it is important to know when to stop. If we have a collection of transcribed utterances, we can track the keyword error rate on that data and halt when the increment in performance is sufficiently small. Keywords here refer to the initial vocabulary.

If there is no transcribed data, then we cannot directly measure the error rate. We can however bound the rate at which it is changing by comparing keyword locations in the output of the recognizer between iterations. If few keywords are shifting location, then the error rate cannot be changing above a certain bound. We can therefore place a convergence criterion on this bound rather than on the actual keyword error rate. It is important to just measure changes in keyword locations, and not changes in vocabulary items added by clustering.

9.4 Offline vocabulary extension

The unsupervised procedure described in the previous section is intended to both improve recognition accuracy on the initial vocabulary, and to identify candidates for vocabulary extension. This section describes experiments that demonstrate to what degree these goals were achieved. To facilitate comparison of this component with other ASR systems, results are quoted for a domain called LCSInfo (Glass and Weinstein, 2001) developed by the Spoken Language Systems group at MIT. This domain consists of queries about personnel – their addresses, phone numbers etc. Very preliminary results for Kismet-directed speech are also given.

Results given here are from a clustering session with an initial vocabulary of five keywords (`email, phone, room, office, address`), run on a set of 1566 utterances. Transcriptions for the utterances were available for testing but were not used by the clustering procedure. Here are the top 10 clusters discovered on a very typical run, ranked by decreasing frequency of occurrence:

```
1n ah m b er   6    p l iy z
2w eh r ih z   7    ae ng k y uw
3w ah t ih z   8    n ow
4t eh l m iy   9    hh aw ax b aw
5k ix n y uw   10   g r uw p
```

These clusters are used consistently by the recognizer in places corresponding to: “number, where_is, what_is, tell_me, can_you, please, thank_you, no, how_about, group,” respectively in the transcription. The first, `/n ah m b er/`, is very frequent because of phrases like “phone number”, “room number”, and “office number”. Once it appears as a cluster the language model is immediately able to improve recognition performance on those keywords.

Every now and then during clustering a “parasite” appears such as `/dh ax f ow n/` (from an instance of “the phone” that the recognizer fails to spot) or `/iy n eh l/` (from “email”). These have the potential to interfere with the detection of the keywords they resemble acoustically. But as soon as they have any success, they are detected and eliminated as described earlier. It is

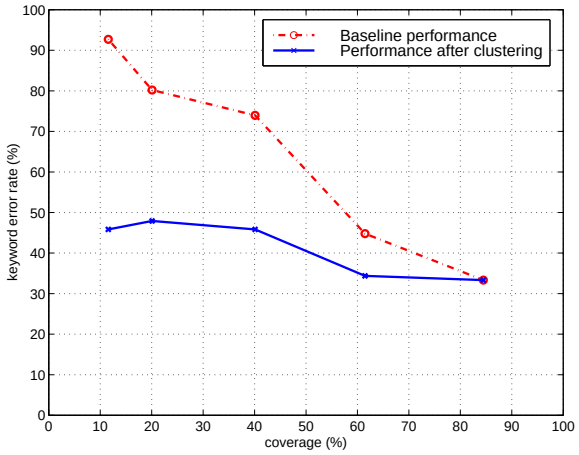


Figure 9.3: Keyword error rate of baseline recognizer and clustering recognizer as total coverage varies.

possible that if a parasite doesn't get greedy, and for example limits itself to one person's pronunciation of a keyword, that it will not be detected, although we didn't see any examples of this happening.

For experiments involving small vocabularies, it is appropriate to measure performance in terms of Keyword Error Rate (KER). Here this is taken to be:

$$KER = \frac{F + M}{T} * 100 \tag{9.1}$$

with:

F	=	Number of false or poorly localized detections
M	=	Number of missed detections
T	=	True number of keyword occurrences in data

A detection is only counted as such if it occurs at the right time. Specifically, the midpoint of the hypothesized time interval must lie within the true time interval the keyword occupies. We take forced alignments of the test set as ground truth. This means that for testing it is better to omit utterances with artifacts and words outside the full vocabulary, so that the forced alignment is likely to be sufficiently precise.

The experiments here are designed to identify when clustering leads to reduced error rates on a keyword vocabulary. Since the form of clustering addressed in this paper is fundamentally about extending the vocabulary, we would expect it to have little effect if the vocabulary is already large enough to give good coverage. We would expect it to offer the greatest improve-

ment when the vocabulary is smallest. To measure the effect of coverage, a complete vocabulary for this domain was used, and then made smaller and smaller by incrementally removing the most infrequent words. A set of keywords were chosen and kept constant and in the vocabulary across all the experiments so the results would not be confounded by properties of the keywords themselves. The same set of keywords were used as in the previous section.

Clustering is again performed without making any use of transcripts. To truly eliminate any dependence on the transcripts, an acoustic model trained only on a different dataset was used. This reduced performance but made it easier to interpret the results.

Figure 9.3 shows a plot of error rates on the test data as the size of the vocabulary is varied to provide different degrees of coverage. The most striking result is that the clustering mechanism reduces the sensitivity of performance to drops in coverage. In this scenario, the error rate achieved with the full vocabulary (which gives 84.5% coverage on the training data) is 33.3%. When the coverage is low, the clustered solution error rate remains under 50% – in relative terms, the error increases by at most a half of its best value. Straight application of a language model gives error rates that more than double or treble the error rate.

As a reference point, the keyword error rate using a language model trained with the full vocabulary on the full set of transcriptions with an acoustic model trained on all available data gives an 8.3% KER.

An experiment was carried out for data drawn from robot-directed speech collected for the Kismet robot. This data comes from an earlier series of recording sessions for the work described in (Breazeal and Aryananda, 2000). Semantically salient words such as “kismet”, “no”, “sorry”, “robot”, “okay” appeared among the top ten clusters.

9.5 Real-time vocabulary extension

In actual operation, ideally new vocabulary items could be added instantaneously, rather than extracted through a slow offline procedure. To achieve this, the robot was given a much simpler vocabulary extension mechanism, where novel isolated words are mimicked back immediately by the robot and added to its lexicon. As words are heard, they are grouped based on a weak measure of similarity – the statistics of pairs of phonemes two words have in common (See Figure 9.4). Similar-sounding words will be merged, but this can be accepted since it would be difficult to reliably differentiate them anyway. In a sense, the robot will only permit sufficiently different sounds to converge to a vocabulary. This method is appropriate if the desired working

"destroy"	"green"	"landmine"	"robot"	"spaghetti"	"yellow"
[d ih s t r ao]	[g r iy n]	[l ae d m ay n]	[r ow b ao]	[s p ix g eh t iy]	[y eh l aw]
d ih s t r oy	g r iy n	n ae n s m ay n	r ow b ao n	t ax g eh t iy	y eh n l ow
d ih s t r ay	g r iy n	l ae d m ay n	r ow b ao	s p ix g eh t iy	y ae l ow
s t r ao	g r iy d	l ae n m ay n	r ow b aw	s p iy t ax	y eh l ow
dh ax s t r ao	r iy n	l ae n m ay n	m ow b ao	d ix g eh	y ax l aw
dh ax s t r oy	d r iy n	l ae d m ay n	r ow v ae	d ix g ih	y eh l aw
d ih s t r ao	g r iy	m ae d m ay n	r aw b ao	s p ix g eh d t iy	
d ey s t r ao d	g r iy	l ae d s m ay n	r ow b aa		
dh ey s t r ao	g r iy n	l ae d m ay n	r ow b aa		
d ih s t r ao	g r iy d	l ah n n ay	r ow b ah		
d ih s t r ay	g r iy d	l ae n t w ay n	r ow w ae		
	k r iy n	n ae n d ix n l ay n			
	r iy	b l ae n t w ay n			

Figure 9.4: Converging on a vocabulary. The top row shows English words that were spoken to the robot repeatedly. The second row shows the phonemic version of those words the robot chose. The remaining rows show the transcripts of each individual utterance. The version chosen by the robot is what it *speaks*, however it will recognize words close to any of the variants as corresponding to the same word. So if the person says "spaghetti" and the robot hears [d ix g ih], then it will recognize and mimic that word as [s p ix g eh t iy]. Clearly this will limit the size of the robot's vocabulary, but that seems a necessary trade-off with the current state of the art.

vocabulary at any point has a relatively small number of words. This is all that can really be supported in a noisy environment without user-specific training or well-placed microphones, anyway. Initially this behavior was achieved by using the dynamic vocabulary API of IBM ViaVoice. It proved simpler to use raw phonemic recognition and an external Viterbi alignment procedure, although ideally this would be merged with the speech recognition system for optimal performance.

9.6 Stabilized perceptual interface

Just like the object recognition system discussed in Chapter 5, recognized words were communicated to the rest of the system using a stabilized interface. As vocabulary items are created, they are assigned a unique 'feature line.' The meaning of that feature line is conserved as much as possible from then on, so that the line will respond to the same situations in future as it did in the past. Offline clustering is done to refine the online vocabulary grouping. This is initially done without any regard to the stabilized interface so that off-the-shelf clustering algorithms can be used; then as a last step models are compared with previous models and aligned appropriately.

CHAPTER 10

Towards interpersonal perception

> *"What is this thing, anyway?" said the Dean, inspecting the implement in his hands. "It's called a shovel," said the Senior Wrangler. "I've seen the gardeners use them. You stick the sharp end in the ground. Then it gets a bit technical."* (Pratchett, 1991a)

Harnad (2002) argues that creatures can learn about categories of objects or other entities either through toil or theft. Sensorimotor toil is his term for "trial-and-error learning, guided by corrective feedback from the consequences of miscategorisation." This is inherently expensive in terms of time and risk. Linguistic 'theft', on the other hand, allows categories to be passed on from other individuals, at a much reduced cost to the recipient. Harnad is mostly concerned with arguing that it can't be theft all the way down, that there must be some grounding in toil. In this thesis, the robot has so far been doing a lot of toil, so it would be interesting to see if it could start doing some theft.

The goal of this chapter is to build the tools necessary for the robot to familiarize itself with novel activities. Since automatic action understanding is currently very limited, social interaction is used as scaffolding for this learning process, to 'steal' structural information about activities from a cooperative human. Learning about activities is important because they provide tools for exploring the environment. Chapter 3 showed that, with a built-in poking activity, the robot could reliably segment objects from the background

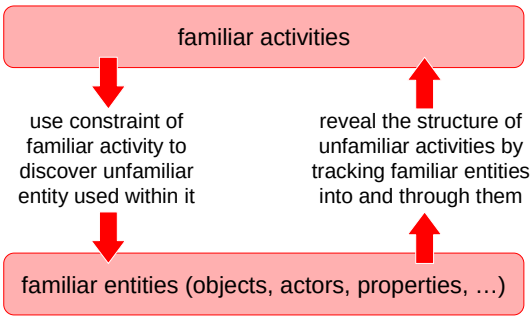


Figure 10.1: If the robot is engaged in a known activity (top), there may be sufficient constraint to identify novel elements within that activity (bottom). Similarly, if known elements take part in some unfamiliar activity, tracking those can help characterize that activity. Potentially, development is an open-ended loop of such discoveries. Familiar activities can be used to learn about components within those activities (for example, the object struck during poking) and then tracked out into novel activities; then when the robot is familiar with those activities it can turn around and use them for learning also.

(even if it is similar in appearance) by poking them. It could determine the shape of an object boundary in this special situation, even though it cannot do this normally. This is the desirable feature of activities for learning – they provide special enabling contexts. In fact, they are key to creating an open-ended developmental cycle (see Figure 10.1). Particular, familiar situations allow the robot to perceive something about objects or object properties that could not be perceived outside those situations. These objects or properties can be tracked into other, less familiar situations, which can then be characterized and used for further discovery. Just as the segmented views provided by poking of objects and actors by poking can be collected and clustered as discussed in Chapter 5, provided precisely what was needed to train up an object detection and recognition system, tracking those objects provides exactly what is needed to learn about other activities, which in turn can be used for further learning.

10.1 Learning through activity

The intersection of communication, perception and development encompasses some well-established fields of research – for example, language acquisition. It has been observed that language acquisition involves a search through a large search space of models guided by relatively sparse feedback and few

examples. This so-called "poverty of the stimulus" relative to the complexity of the models being acquired is taken to imply that infants must have a good search strategy, with biases well matched to the nature of appropriate solution. This is a claim of innate constraints, and is historically controversial. Examples stressing under-determination in language learning include Quine's "Gavagai" example (Quine, 1960), where Quine invites us to imagine ourselves walking with a native guide in a foreign country, and seeing a rabbit pass just as the guide says "gavagai" – and then consider all the possible meanings this utterance might have.

Pragmatic constraints offer one way out of this sea of ambiguity. For example, Markman (1989) proposes a set of particular constraints infants might use to map words on to meanings. These constraints are along the style of the following (with many variations, elaborations and caveats) :-

- ▷ Whole-object assumption. If an adult labels something, assume they are referring to the whole object and not a part of it.
- ▷ Taxonomic assumption. Organize meanings by "natural categories" as opposed to thematic relationships. For example when a child is asked to find a "dog", he/she may fetch the cat, but won't fetch dog-food.
- ▷ Mutual exclusivity. Assume objects have only one label. So look for an unnamed object to which a new label can be applied.

These constraints are intended to explain a spurt in vocabulary acquisition where infants begin to acquire words from one or a few examples – so-called fast-mapping. They are advanced not as absolute rules, but as biases on search.

Tomasello raises several objections to the constraint-based approach represented by Markman Tomasello (1997). Tomasello favors a "social-pragmatic" model of language acquisition that places language in the context of other joint referential activity, such as shared attention. He rejects the "word to meaning mapping" formulation of language acquisition. Rather, Tomasello proposes that language is used to invite others to experience the world in a particular way. From Tomasello (1997) :-

> The social-pragmatic approach to the problem of referential indeterminacy ... begins by rejecting truth conditional semantics in the form of the mapping metaphor (the child maps word onto world), adopting instead an experientialist and conceptualist view of language in which linguistic symbols are used by human beings to invite others to experience situations in particular ways. Thus, attempting to map word to world will not help in situations in which the very same piece of real estate may be called: "the shore" (by a sailor), "the coast" (by a hiker), "the ground" (by a skydiver), and "the beach" (by a sunbather).

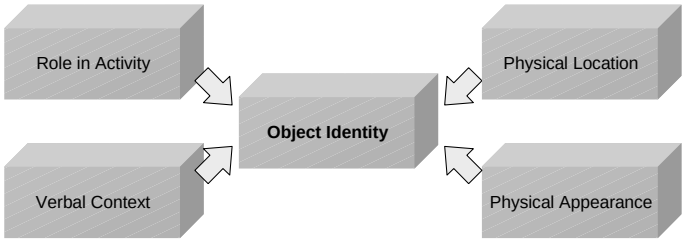


Figure 10.2: Perceptual judgements are fundamentally about identity: what is the same, what is different. Identity judgements should depend (at least) on activity, location, appearance, and verbal context. These in turn can be influenced by a teacher.

Regardless of the utility of Tomasello's theory for its proper domain, language acquisition in infants, it seems a useful mindset for tackling interpersonal perception, which is in essence all about inviting the robot to view the world in a particular way.

Tomasello and his collaborators developed a series of experiments designed to systematically undermine the constraints approach to learning as typified by Markman and others. The experiments investigate word learning among children in the context of various games. The experiments are instructive in showing a range of situations in which simple rules based directly on gaze or affect would fail in at least one case or other. The experiments all avoid giving children (18-24 months old) ostentive naming contexts, and rather requiring them to pull out meanings from the "flow of interaction".

For example, in one experiment, an adult makes eye-contact with a child subject and says "Let's go find the toma," where toma is a nonsense word the child has never heard before. They then go to a row of buckets, each if which contains an object with which the child is not familiar. One of these objects is randomly designated the "toma". If the session is a control, the adult goes directly to the bucket containing the toma, finds it excitedly and hands it to the child. Otherwise, the adult first goes to two other buckets in sequence, each time taking out the object, scowling at it, and replacing it, before "finding" the toma. Later, the child is tested for for the ability to comprehend and produce the new word appropriately. The results show equally good performance in the test and control scenarios. Tomasello argues that this situation counts against children using simple word learning rules such as "the object the adult is looking at while saying the novel word," "the first new object the adult looks at after saying the novel word," "the first new object the infant sees after hearing the novel word," or such variants.

Tomasello's theories and experiments are provocative, and suggest an ap-

proach quite different from the simple associative learning that is most often seen in robotics. Work on interpersonal perception on Cog draws heavily on (a grossly simplified caricature of) these ideas. The basic idea for interpersonal perception drawn from Tomasello's work is that information about the identity of an object needs to be easily transferred between perception of activity, location, speech, and appearance (Figure 10.2). Without this flexibility, it is hard to imagine how scenarios such as the experiment described above or others proposed (Tomasello, 1997) could be dealt with.

10.2 Places, objects, and words

It is currently unreasonable to expect a robot to understand a "flow of interaction" without help. Unaided segmentation of activity is a very challenging problem (see Goldberg and Mataric (1999) for one effort in the robotic domain). The human interacting with the robot can greatly simplify the task by making the structure of the activity unambiguous. Two mechanisms for this are particularly easy to deal with: vocalizations and location. If places and words are used consistently in an activity, then it is straightforward to model the basic "flow of interaction" they define.

The robot was given a verbal, 'chatting' behavior to augment the object-directed poking behavior developed in 3. This used the vocabulary extension mechanism developed in Chapter 9, the egocentric map developed in Chapter 8, and the ability to recognize poked objects developed in Chapter 5. If the robot hears a word while fixating a particular object, and that word has not been heard in other context, then the word is associated with the object. If this happens several (three) times, the association is made permanent for the session. Invocation of an object by name triggers the egocentric map to drive the eyes to the last known location of the object, and the foveal object recognition module to search for the object visually (see Figure 10.3).

This simple naming capability serves as a baseline for the rest of this chapter, which will show how the robot can learn new opportunities for associating names and objects in situations without ostentive showing.

10.3 Learning the structure of a novel activity

Before the robot can learn through an activity, it must be able to learn about that activity. If it cannot do this, then it will remain restricted to the set of built-in activities provided by the programmer. Ideally, it should be possible to demonstrate a task to a robot and have it learn to do it. As already mentioned, unaided segmentation of activity is a very challenging problem

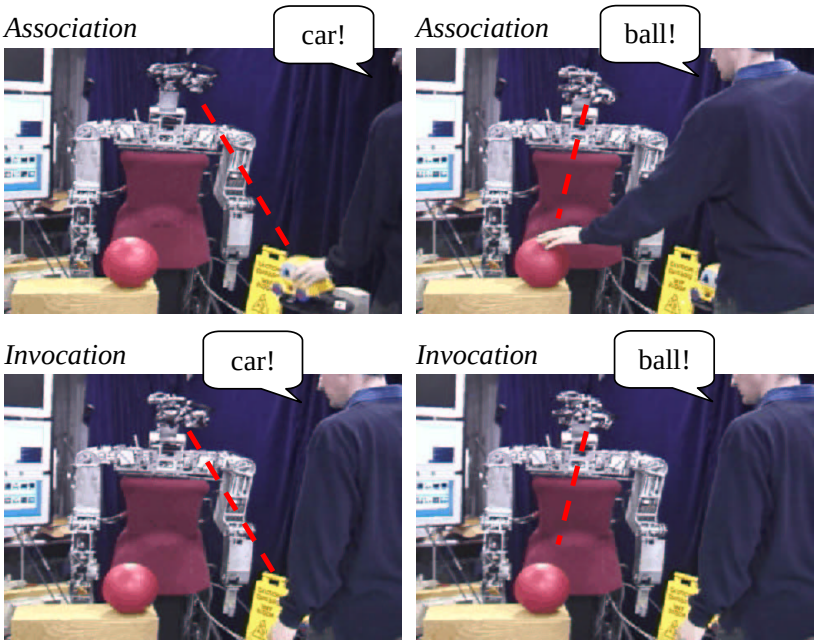


Figure 10.3: Association and invocation via the egocentric map. When the robot looks at an object and recognizes it, its head rolls into an inquisitive look. If a word is spoken at this point (e.g. “car!” or “ball!” in top two frames – note that the human is bringing the robot’s attention to an object with his hand) then that word is associated with the object the robot is viewing. If that word is spoken again later (as in the lower frames – note that the human is standing back, only interacting through speech), then the robot queries the egocentric map for the last known location of the associated object, turns there, and looks for the object.

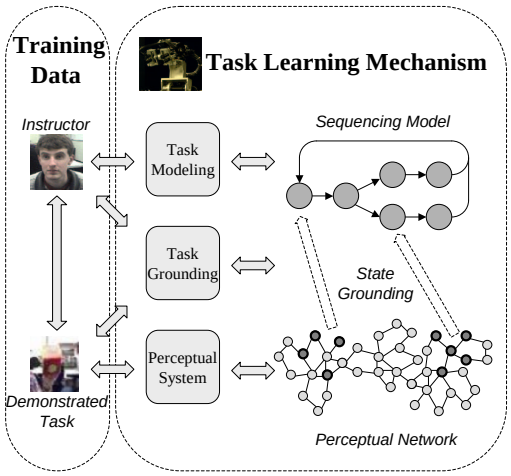


Figure 10.4: A model of task segmentation. The instructor demonstrates the task while providing verbal annotation. The vocal stream is used to construct a model of the task. Generic machine learning methods are then used to ground this model in the robot's perceptual network, guided by feature selection input from the human. The idea is to avoid ever presenting the robot with a hard learning problem; the learning algorithms are intended to be "decoders" allowing the human to communicate changes in representation, rather than to learn in the conventional sense.

in machine perception. It is perhaps more productive to see activity segmentation as something that is explicitly communicated to the robot, rather than something it learns autonomously.

While individual parts of a task may be difficult to describe formally, its abstract structure or control flow will often be amenable to simple description. For example, the overall branch-and-loop flow of a sorting task is easily expressed, but the actual sorting criterion may depend on differentiating two classes of objects based on a small difference in their appearance that would be easier to demonstrate than to describe. If we go ahead and communicate the task structure to the robot, it can be used to guide interpretation of the less easily expressed components. Figure 10.4 shows a schematic for how this may be achieved. The basic idea is for the robot to interact with the instructor vocally to acquire a "sequencing model" of that task, and then to ground that model based on a demonstration of the task. The demonstration is annotated by the instructor both in terms of the sequencing model and in terms of previously grounded elements.

As the human tutor demonstrates a task, they are expected to verbalize

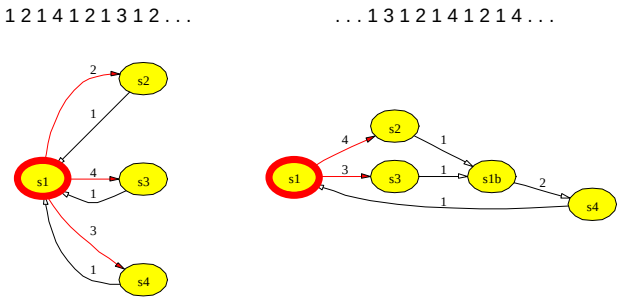


Figure 10.5: The top row of this figure shows a simple artificial sequence presented to the task segmentation system. For the purposes of visualization, the current estimate of the task structure is converted to a state machine and rendered automatically using AT&T Graphviz (Gansner and North, 2000). Initially, the sequence is interpreted as alternating between 1 and any of 2, 3, or 4 (left). Once more data is available, the model expands to incorporate the fact that there is a second-order alternation between 2 and either 3 or 4 (right).

their activity. Initially the robot cannot make much of the physical demonstration, but it can process the speech stream, and attempt to recover the structure of the task from that. In particular, the robot will attempt to determine "states" of the task – points at which the demonstration returns to what is effectively the same mode. This is straightforward to do if the human tutor speaks simple vocal labels corresponding to actions, configurations, objects, or whatever the tutor finds mnemonic. The type of labels used does not need to be pre-specified, and could vary from word to word.

The method used to recover the task structure is based on n-gram modeling procedures developed for speech recognition – although there are many other methods (Murphy, 1996), this one was chosen for its simple, easily predicted behavior. Here, we estimate the probability of event sequences from models trained on sequence frequency counts from a corpus. Models vary in the amount of history they incorporate – bigram models, trigram models etc. Low order models are limited in the dependencies they can capture, but can be trained up with relatively little data. High order models are more expressive but harder to train. Best results are achieved when n-gram models of many different orders are used, and interpolated based on the amount of training data available for each context (see Figure 10.5 for a simulated example). Once the robot has a model for the task structure, the goal is to relate that to the actual physical demonstration the human tutor is making. Machine perception is necessarily noisy and full of ambiguity. The degree to which this

Human speech	Human action	Robot speech	Robot action
…	…	…	…
say yellow	[shows yellow toy]	yen	[looks at toy]
say yellow		yelo	
yes! say green		green	
…	…	…	…
yellow	[shows yellow toy]	yelo	[looks at toy]
left	[moves toy left]	left	[tracks toy]
green	[shows green toy]	green	[looks at toy]
right	[moves toy right]	right	[tracks toy]
yellow	[shows yellow toy]	yelo	[looks at toy]
left	[moves toy left]	left	[tracks toy]
…	…	…	…
yellow	[shows yellow toy]	yelo	[looks at toy]
next?		ahm left?	[nods left]
yes!	[moves toy left]	left!	
green	[shows green toy]	green	[looks at toy]
next?		ahm right?	[nods right]
yes!	[moves toy right]	right!	
…		…	

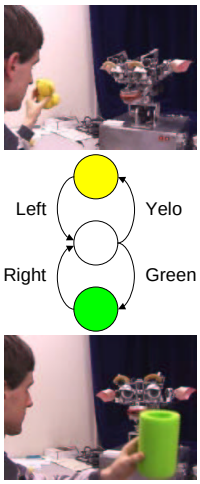


Figure 10.6: Extracts from a dialogue with Kismet. The first extract (say yellow...) illustrates how the robot's active vocabulary was extended. On Cog, this is replaced with the automatic mechanism described in Chapter 9. The second extract shows how a simple sorting activity was annotated for the robot. The final extract shows the robot being tested on its understanding of the form of the activity. The robot's utterances were transcribed phonetically, but are written in a simple form here for clarity. To the right is shown the simple state machine model of the activity deduced by the robot (graph drawn by hand).

is so for a given task will fundamentally limit the complexity of any modeling the robot can do, if we permit uncertainty to compound on uncertainty. By first establishing a task model through a relatively noise-free protocol for which we can depend on error-correcting feedback from the human tutor, we limit the impact that uncertainty in grounding one element of the model will have on all the others.

Figure 10.6 shows an example of this for a sorting activity, implemented on the robot Kismet. Note that words are used here without the robot needing to know their meanings – it is sufficient that they be used consistently enough for the structure of the task to be made obvious.

10.4 Learning the rules of a novel activity

The structure learning mechanism in the previous section is useful, but if a person interacting with the robot is cooperative in their choice of vocabulary, it is in fact overkill. It also does not deal well with nested activities, where one activity is suspended temporarily to deal with another – for example, if during the sorting behavior the person wants to check if the robot knows the name of an object. When transferring the mechanism from Kismet to Cog, the emphasis was shifted from learning global task structure to learning local rules (which could grow to support long-distance interactions). On Cog, seeing objects or hearing words are treated as the basic events in the system. The robot continually searches for useful new ways to describe events, where being 'useful' means having predictive power. The events it considers are :-

- ▷ **Conjunctions:** if two events are noted to occur frequently together, and rarely occur without each other, an event called their conjunction is formed. This event is defined to occur whenever the two events do in fact occur together. The formation of an event simply means that statistics related to it are tracked. Once an event is formed, it doesn't matter if the conditions for its creation cease to hold.
- ▷ **Disjunctions:** if two events are noted to occur frequently together, but also occur independently in other situations, an event called their disjunction is formed. This event is defined to occur whenever one or both of the two original events occur.
- ▷ **Implications:** Causal versions of the above events also exist, which are sensitive to event order and timing.

These composite events are intended to allow the robot to make meaningful generalizations, by allowing the same physical event to be viewed in ways that are sensitive to past history. Figure 10.7 demonstrates the use of such generalizations to solve one of Tomasello's experiments – linking an object with its name through an extended search activity. Searches are presented to the robot as following a fairly strict script: first the word 'find' is uttered, then the name of the object to search for is mentioned. Then a series of objects are fixated. The word 'no' is uttered if the object is not the target of the search. The word 'yes' indicates that the search has succeeded, and the object currently fixated is the target of the search. These facts can be discovered using event generalization. The word spoken after 'find' gets a special implication event associated with it, let us call it `word-after-find` (of course, no such symbols are used internally, and the word 'find' initially has no special significance – it could be replaced with any other word, such as 'seek,' 'cherchez,' or 'fizzle-tizzle'). If the search is for an object whose name the robot knows (through a pre-established disjunction) that is

also noted as a simultaneous event with `word-after-find`. The object seen when 'yes' (`object-with-yes`) is said matches this and an implication is formed between the two. This implication is sufficient to link an *unknown* word following 'find' with the object seen when 'yes' is said, via the `word-after-find` and `object-with-yes` generalizations (again, the choice of the word 'yes' has no special significance, and could be replaced with 'frob').

When the rule-learning mechanism above was integrated with the full Cog code-base, then the search activity became much simpler to learn, requiring less generalization. This is because for the 'chatting' behavior, the robot has internal state to track when it hears a name of an object it is not looking at (so that it can then direct its gaze towards that object). With this structure built in, the robot simply has to map the search activity on to it, which it can do with just two observations:

- ▷ 'Find' is followed by mention of an absent object
- ▷ 'Yes' is said when a previously absent object is in view

The generalizations already built in to the robot's state are sufficient to achieve the first Tomasello scenario.

Human speech	Human action	Robot speech	Robot action
…	…	…	…
say	[shows ball]	say	[looks at ball]
beh		ball	
say	[shows car]	say	[looks at car]
keh		car	
say	[shows cube]	say	[looks at cube]
keh		cube	
say		say	
	[waits]	cube	
	[shows ball]		[looks at ball]
say		say	
	[waits]	ball	
…	…	…	…
	[attracts attention]		[looks at person]
find		find	
ball		ball	
no	[shows cube]	no	[looks at cube]
no	[shows car]	no	[looks at car]
yes	[shows ball]	yes	[looks at ball]
…	…	…	…
	[attracts attention]		[looks at person]
find		find	
toma		toma	
no	[shows ball]	no	[looks at ball]
no	[shows cube]	no	[looks at cube]
yes	[shows bottle]	yes	[looks at bottle]
say	[shows cube]	say	[looks at cube]
		cube	
say	[shows bottle]	say	[looks at bottle]
		toma	
…	…	…	…

Figure 10.7: Extracts from a dialogue with Cog. First, the robot is taught to name the object it is looking at when the word ‘say’ is spoken. This is done by speaking the word, then prompting the robot with a short utterance (beh and keh in this example). Short utterances prompt the robot to take responsibility for saying what it sees. A link is formed between ‘say’ and prompting so that ‘say’ becomes an alternate way to prompt the robot. Then the robot is shown instances of searching for an object whose name it knows (in the one example given here, the ball is the target). Finally, the robot is shown an instance of searching where an unfamiliar object name is mentioned (‘toma’). This allows it to demonstrate that it has learned the structure of the search task, by correctly linking the unfamiliar name (‘toma’) with the target of search (a bottle). Ideally, to match Tomasello’s experiment, all the objects in this search should be unfamiliar, but this was not done. In the infant case, this would leave open the possibility that the infant associated the unfamiliar word with the first unfamiliar object it saw. In the robot case, we have access to the internal operations, and know that this is not the cue being used.

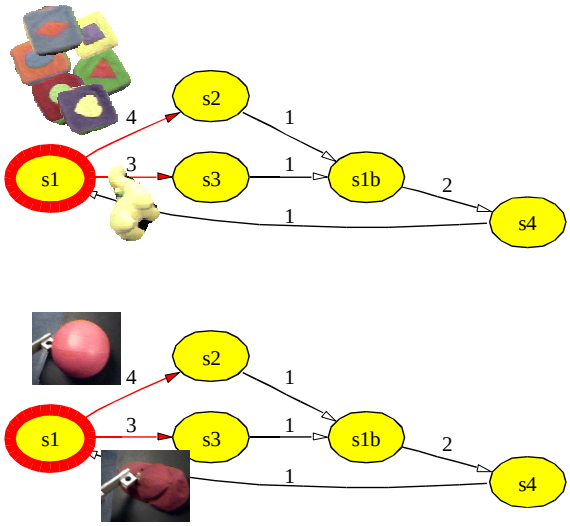


Figure 10.8: Task structure should allow the robot's perceptual biases to be overruled. for example, objects are differentiated by the robot purely based on color histogram. This could cause problems for an object which looks very different from different sides (such as a toy cube, top). These views could be united within a task where they are all treated the same way (for example, by performing one action on the cube and another on another toy). If two distinct objects are treated as the same by the robot because of color similarity (such as a ball and baseball cap, bottom), then their difference can be exaggerated by using them differently within a task.

10.5 Limitations and extensions

There are many limitations to the activity learning described in this chapter, including :-

- The cues the robot is sensitive to are very impoverished, relative to what a human infant can perceive. For example, there is no direct representation of the teacher, and no perception of prosody or non-verbal cues.
- If multiple activities share similar vocabularies, their is the potential for interference between them. The issue of capturing the overall activity context has not been addressed.
- The basic events used are word and object occurrences, which do not begin to capture the kind of real world events that are possible. So the

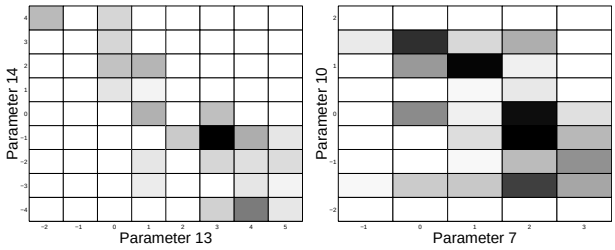


Figure 10.9: Searching for correlations in the robot's perceptual features. The joint probability distribution of pairs of features is estimated, and compared with the product of their independent distributions. This figure shows the top two correlations in an experiment where the robot applied one of four actions to an object (tapping it from the side, slapping it away, etc.). The highest ranking correlation, left, captures a physical constraint on the angle of approach at the moment of impact. The next correlation (right) captures the gross displacement of the object away from the robot as a function of the type of gesture (the correlation is noisy because of erratic motor control). With verbal annotation of the actions, this correlation could be enhanced (by tagging failed actions for removal) and selected out for use.

> robot could not respond to non-speech sounds, or changes in distance, or any of the infinite possible events that are not simply word/object appearances.

To begin to deal with this last point, a simple mechanism was developed to get the robot's attention to an unnamed feature or feature combination (as opposed to simply an object) using periodicity detection. All perceptual features on Cog are monitored over a sixty second time window to detect the occurrence of periodicity. Hence if it is desired that the robot attend to the color of objects as opposed to their identity or size, for example, then objects of contrasting colors can simply be shown to the robot. The periodic signal oscillation increased the salience of a channel in a manner similar to the behavioral influences used on Kismet (Breazeal and Scassellati, 1999). But at the time of writing, this was not strongly integrated with the activity learning mechanisms.

The idea of influencing the robot's perception through shared activity also could and should be developed further. Perception is not a completely objective process; there are choices to be made. For example, whether two objects are judged to be the same depends on which of their many features are considered essential and which are considered incidental. For a robot to be useful, it should draw the same distinctions a human would for a given task. To achieve

this, there must be mechanisms that allow the robot's perceptual judgements to be channeled and molded by a teacher. This would also useful in situations where the robot's own abilities are simply not up to the challenge, and need a helping hand. Once the structure of tasks can be communicated to the robot, it should be possible to use that high-level structure to modify the robot's perception. It is easiest to see this in the case of modifying biases in pre-existing abilities of the robot. For example, we could emphasize the difference between objects the robot sees as identical, or draw connections between different views of the same object that the robot sees as distinct (see Figure 10.8).

More generally, we can use the task structure to initialize a set of focused problems in machine learning, where divergent paths in the task are treated as labels for the (initially unknown) physical features that cause that divergence. By correlating features across the robot's perceptual space with these labels, we can select those that might contribute to the decision, and then train up a classifier. Figure 10.9 shows an example of searching for such correlations in the robot's perception of its own poking behavior.

10.6 Summary

This chapter has shown that a virtuous circle of development is possible (see Figure 10.1. If we want robots to be able to cope with novel tasks, they will need a deep understanding of the activities around them. We can treat the range of naming situations a robot can deal with as a test of the depth of that understanding. Consider search: if the robot understands the purpose of searches, how they succeed and fail, then that naturally extends the range of naming situations it can deal with beyond simple ostentive associations, as this chapter showed. In the infant development literature, considerable emphasis is placed on the child's ability to interpret the behavior of others in terms of intent using a "theory of mind". Such an ability is very powerful, but so also is the more computational viewpoint of processes as branches, loops and sequences. This is an alternative route to establish an initial shared perspective between human and robot, and could potentially complement a theory of mind approach (for example, the work of Scassellati (2001) on Cog).

In the activity-learning system described here, the meanings of words grow out of their role within an activity. In the search activity, 'yes' and 'no' come to denote the presence or absence (respectively) of the search target. In another scenario, they may denote something entirely different. For example, it would be possible to train the robot to keep holding out its arm while 'yes' is said, and to drop it upon hearing 'no' – in which case the words now denote action continuance or termination respectively. This plasticity means that we

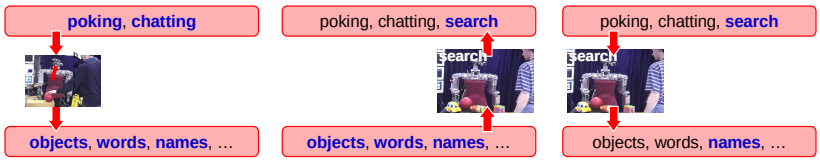


Figure 10.10: The chapter developed a specific example of the virtuous circle of development. First, poking allows the robot to explore objects, and then chatting allows names to be associated with those objects. Then the robot tracks those named objects as a human demonstrates a search task, learning about the structure of search from these examples. Finally, the robot uses this knowledge as a new way to learn names for objects without having to see an object and hear its name simultaneously, as is the case for chatting.

can avoid the problem of trying to form a global theory of all the meanings a word can take on.

Richer meanings are possible when multiple-word utterances are permitted (Roy et al., 2002, 2003), rather than the isolated words dealt with in this chapter. An interesting direction for future research would be to derive grammatical forms as a compression of the structure of an extended activity into a single sentence.

CHAPTER 11

Conclusions and future directions

> *It's always worth having a few philosophers around the place. One minute it's all Is Truth Beauty and Is Beauty Truth, and Does a Falling Tree in the Forest Make a Sound if There's No one There to Hear It, and then just when you think they're going to start dribbling one of 'em says, "Incidentally, putting a thirty-foot parabolic reflector on a high place to shoot the rays of the sun at an enemy's ship would be a very interesting demonstration of optical principles.*" (Pratchett, 1992b)

The work in this thesis was motivated by fallibility and transience. As humanoid robots become more sophisticated mechanically, it seems likely that their perceptual abilities will become a severe limiting factor on what they can do. In the absence of perfect perception, it will be important to have simple experimental methods that resolve ambiguity, and methods to derive information from clumsy actions so they are not repeated. Active segmentation is a good example. We may also expect that humanoid robots will need considerable flexibility – perhaps the task for the robot may change from day to day. It is best to build in adaptivity from very beginning. This thesis has made some steps towards building a perceptual system for a robot that can grow and develop through contact with the world. This is both a theoretical effort to show, for example, how adaptive modules can have persistent interfaces, and a practical effort of identifying and engineering opportunities for the robot to develop (see Figure 11.1). The areas explored are for the

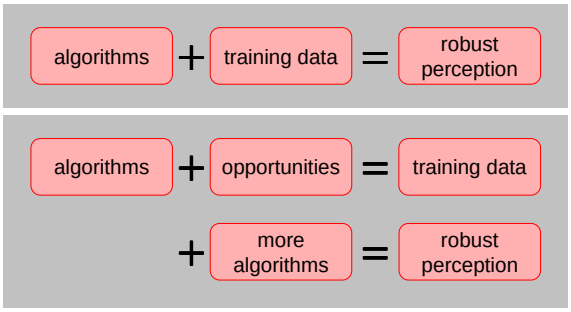


Figure 11.1: One route to autonomy is to switch the emphasis from collecting training data (top) to engineering methods to create and exploit opportunities for collecting and labeling such data autonomously.

most part complementary to other work on development in robotics (Metta, 2000; Roy and Pentland, 2002; Weng et al., 2000b). This chapter summarizes the specific contributions made and identifies some important directions for future research.

11.1 Summary of significant contributions

- **active segmentation** – the thesis showed that passive methods for object segmentation can be augmented by physical exploration.
- **appearance catalog** – the appearance of small, low-level features can be almost exhaustively characterized, as this thesis showed for oriented features, given results that are competitive with classical model-driven approaches.
- **open object recognition** – for a robot it is important to integrate object recognition with a mechanism for enrolling new objects, since there are far more objects in the world than can reasonably be catalogued. This thesis showed that this makes false detections a less serious problem, because distracting objects can simply be enrolled and modelled explicitly, rather than having to come up with an accurate background model.
- **affordance recognition** – for a robot, it makes sense to switch from object-centric perception to recognizing action opportunities. A concrete example of this is given for rolling, an affordance that is of particular importance to a robot that needs to manipulate awkward objects. This work was a collaboration with Giorgio Metta.
- **open speech recognition** – in speech recognition, there is a trade-

off between recognition accuracy and vocabulary size. This thesis assumes that at any time the vocabulary a robot needs is small and task-dependent. By creating explicit run-time mechanisms for vocabulary modification, the robot can quickly be given the vocabulary appropriate to the current task without needing a large pre-existing vocabulary.

- ▷ **virtuous circle of development** – familiar activities can be used to identify components used in roles within those activities. Then those components can be tracked out into unfamiliar activities, and used to discover the structure of those activities. These two processes dovetail to give a circle of development.

11.2 Grounding operational definitions

In this thesis, the appearances of objects and manipulators were characterized by using operational definitions. These are definitions which translate to measurements :-

> An operational definition is a procedure agreed upon for translation of a concept into measurement of some kind. – (Deming, 1993)

An effective procedure for finding objects, seen as physically coherent structures, is to poke around and see what moves together. An effective procedure for finding manipulators, defined as something that acts on objects, is to watch what pokes objects. Of course, both these procedures are not completely general, and they are worth generalizing. For example, active segmentation gives clear results for a rigid object that is free to move, but what happens for non-rigid objects and objects that are attached to other objects? Here the results of poking are likely to be more complicated to interpret – but in a sense this is a good sign, since it is in just such cases that the idea of an object becomes less well-defined. Poking has the potential to offer an operational theory of "object-hood" that is more tractable than a vision-only approach might give, and which cleaves better to the true nature of physical assemblages.

11.3 Fully autonomous platform

Both Cog and Kismet are fixed platforms, and so have access to a very limited environment. A new project in the Humanoid Robotics Group at the AI lab seeks to remedy that (see Figure 11.2). This project combines several

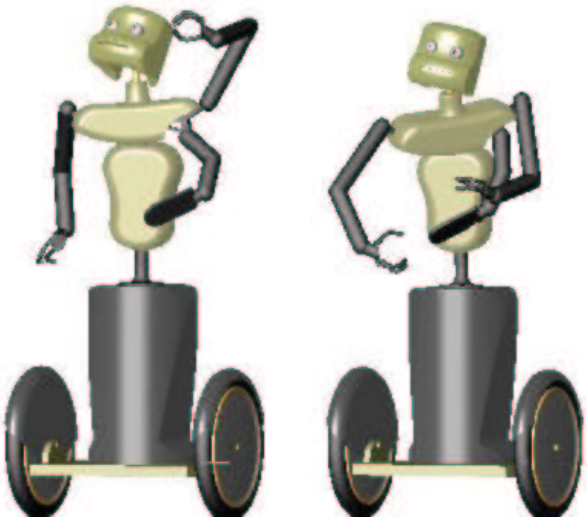


Figure 11.2: A segway-based robot (drafted by Jeff Weber).

important threads: mobility (the robot will have a modified segway base), expressiveness (the robot head has a simple face and eyes), and dexterity (the robot has three arms). This combines all the elements needed for autonomous object exploration and activity learning.

11.4 Philosophy

This thesis has focused primarily on learning perception, and is complementary to other work on Cog that addresses learning action (Marjanović, 2003). In animals and robots, perception is fundamentally for action:

> Perceptions and ideas are found, upon analysis, to have their factual significance and meaning in terms ultimately of overt operation. Their meaning derives from the potential effect, that is, the difference they make or may make in behavior. In both its phylogenetic and ontogenetic histories, mental activity develops out of, and in reference to, overt action. – (Sperry, 1952)

This work is very much influenced by the "alternative essences of intelligence" enumerated in Brooks et al. (1998) – development, social interaction, embodiment, and integration. It attempts to bring all these threads together in one system, and show that they do in fact reinforce each other. Strong

integration of perception and action provides the means for development to occur; social interaction and embodiment provide the opportunities. To those familiar with murder mysteries, the missing element is ‘motive.’ A complete treatment of this topic should examine how to move gracefully between training activity, where the goal is to learn, and actual performance of a task, where the goal is to achieve some more specific end.

Bibliography

Adolph, K. E., Eppler, M. A., and Gibson, E. J. (1993). Development of perception of affordances. *Advances in Infancy Research*, 8:51–98.

Aloimonos, J., Weiss, I., and Bandopadhay, A. (1987). Active vision. *International Journal on Computer Vision*, 2:333–356.

Arsenio, A., Fitzpatrick, P., Kemp, C., and Metta, G. (2003). The whole world in your hand: Active and interactive segmentation. Accepted for publication in the proceedings of the Third International Workshop on Epigenetic Robotics.

Aslin, R., Woodward, J., LaMendola, N., and Bever, T. (1996). Models of word segmentation in fluent maternal speech to infants. In Morgan, J. and Demuth, K., editors, *Signal to Syntax: Bootstrapping From Speech to Grammar in Early Acquisition*. Lawrence Erlbaum Associates: Mahwah, NJ.

Bajcsy, R. (1988). Active perception. *Proceedings of the IEEE*, 76(8):996–1005.

Ballard, D. (1989). Behavioral constraints on animate vision. *Image and Vision Computing*, 7:1:3–9.

Ballard, D. H. (1991). Animate vision. *Artificial Intelligence*, 48(1):57–86.

Bard, E. and Anderson, A. (1994). The unintelligibility of speech to children: effects of referent availability. *Journal of Child Language*, 21:623–648.

Basu, S., Essa, I., and Pentland, A. (1996). Motion regularization for model-based head tracking. In *Proceedings of the International Conference on Pattern Recognition*, Vienna, Austria.

Bazzi, I. and Glass, J. (2000). Modeling out-of-vocabulary words for robust speech recognition. In *Proceedings of the Sixth International Conference on Spoken Language Processing*, Beijing, China.

Beardsley, P. A. (1998). A qualitative approach to classifying head and eye pose. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 208–213, Florence, Italy.

Billard, A. (2001). Imitation: a means to enhance learning of a synthetic proto-language in an autonomous robot. In Dautenhahn, K. and Nehaniv, C. L., editors, *Imitation in Animals and Artifacts*. MIT Press.

Billard, A. and Dautenhahn, K. (1997). Grounding communication in situated, social robots. Technical Report UMCS-97-9-1, University of Manchester.

Birchfield, S. (1998). Elliptical head tracking using intensity gradients and color histograms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 232–237.

Black, M. and Yacoob, Y. (1995). Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion. In *Proceedings of the International Conference on Computer Vision*, pages 374–381.

Block, N. (1978). Troubles with functionalism. *Perception and Cognition: Issues in the Foundations of Psychology, Minnesota Studies in the Philosophy of Science*, 9:261–325.

Bloom, P. (2000). *How Children Learn the Meaning of Words*. Cambridge: MIT Press.

Blumberg, B. (1996). *Old Tricks, New Dogs: Ethology and Interactive Creatures*. PhD thesis, MIT.

Borchardt, G. C. (1993). Causal reconstruction. Technical Report AIM-1403, MIT Artificial Intelligence Laboratory.

Boykov, Y. and Kolmogorov, V. (2001). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 359–374.

Breazeal, C. (2000). *Sociable Machines: Expressive Social Exchange Between Humans and Robots*. PhD thesis, MIT Department of Electrical Engineering and Computer Science.

Breazeal, C. and Aryananda, L. (2000). Recognition of affective communicative intent in robot-directed speech. In *Proceedings of the International IEEE/RSJ Conference on Humanoid Robotics*, Cambridge, MA.

Breazeal, C., Edsinger, A., Fitzpatrick, P., and Scassellati, B. (2001). Active vision for sociable robots. *IEEE Transactions on Systems, Man, and Cybernetics, A*, 31(5):443–453.

Breazeal, C., Edsinger, A., Fitzpatrick, P., Scassellati, B., and Varchavskaia, P. (2000). Social constraints on animate vision. *IEEE Intelligent Systems*, 15:32–37.

Breazeal, C. and Fitzpatrick, P. (2000). That certain look: Social amplification of animate vision. In *Proceedings AAAI Fall Symposium, Socially Intelligent Agents - The Human in the Loop*, North Falmouth, MA, USA.

Breazeal, C. and Scassellati, B. (1999). A context-dependent attention system for a social robot. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1146–1151, Stockholm, Sweden.

Breazeal, C. and Scassellati, B. (2000). Infant-like social interactions between a robot and a human caretaker. *Adaptive Behavior*, 8(1):49–74.

Breazeal, C. and Velasquez, J. (1998). Toward teaching a robot "infant" using emotive communication acts. In *Proceedings of the Simulated Adaptive Behavior Workshop*.

Brent, M. and Siskind, J. (2001). The role of exposure to isolated words in early vocabulary development. *Cognition*, 81:B33–B44.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2:14–23.

Brooks, R. A. (1991a). Intelligence without reason. In *Proceedings of the 1991 International Joint Conference on Artificial Intelligence*, pages 569–595.

Brooks, R. A. (1991b). Intelligence without representation. *Artificial Intelligence Journal*, 47:139–160. originally appeared as MIT AI Memo 899 in May 1986.

Brooks, R. A., Breazeal, C., Irie, R., Kemp, C. C., Marjanović, M., Scassellati, B., and Williamson, M. M. (1998). Alternative essences of intelligence. In *Proceedings of the American Association of Artificial Intelligence*, pages 961–968.

Brooks, R. A., Breazeal, C., Marjanovic, M., and Scassellati, B. (1999). The Cog project: Building a humanoid robot. *Lecture Notes in Computer Science*, 1562:52–87.

Brooks, R. A. and Stein, L. A. (1994). Building brains for bodies. *Autonomous Robots*, 1(1):7–25.

Bullowa, M. (1979). *Before Speech: The Beginning of Interpersonal Communicaion*. Cambridge University Press, Cambridge, London.

Burnham, D., Francis, E., Kitamura, C., Vollmer-Conna, U., Averkiou, V., Olley, A., and Paterson, C. (1998). Are you my little pussy-cat? acoustic, phonetic and affective qualities of infant- and pet-directed speech. In *Proceedings of the Fifth International Conference on Spoken Language Processing*, volume 2, pages 453–456.

Butterworth, G. (1991). The ontogeny and phylogeny of joint visual attention. In Whiten, A., editor, *Natural Theories of Mind*. Blackwell.

Canny, J. F. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):34–43.

Cassell, J., Pelachaud, C., Badler, N., Steedman, M., Achorn, B., Becket, T., Douville, B., Prevost, S., and Stone, M. (1994). Animated conversation: Rule-based generation of facial expression, gesture, and spoken intonation for multiple conversational agents. *SIGGRAPH'94*.

Chabat, F., Yang, G. Z., and Hansell, D. M. (1999). A corner orientation detector. *Image and Vision Computing*, 17(10):761–769.

Chapman, D. (1990). Vision, instruction, and action. Technical Report AITR-1204, MIT AI Laboratory.

Chapman, D. and Agre, P. E. (1987). Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 268–272.

Chen, J., Sato, Y., and Tamura, S. (2000). Orientation space filtering for multiple orientation line segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(5):417–429.

Chen, Q., amd T. Shioyama, H. W., and Shimada, T. (1999). 3D head pose estimation using color information. In *Proceedings of the Sixth IEEE International Conference on Multimedia Computing and Systems*, Florence, Italy.

Cole, R. and Yap, C. (1987). Shape from probing. *Journal of Algorithms*, 8(1):19–38.

Colombetti, M. and Dorigo, M. (1994). Training agents to perform sequential behavior. *Adaptive Behavior*, 2(3):247–275.

Connell, J. (1989). A colony architecture for an artificial creature. Technical Report AITR-1151, MIT AI Laboratory.

Cordea, M., Petriu, E., Georganas, N., Petriu, D., and Whalen, T. (2000). Real-time 2.5D head pose recovery for model-based video-coding. In *Proceedings of the IEEE Instrumentation and Measurement Technology Conference*, Baltimore, MD, USA.

DeCarlo, D. and Metaxas, D. (1996). The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 231–238.

Deming, W. E. (1993). *The New Economics for Industry, Government, and Education*. MIT Center for Advanced Engineering, Cambridge, Massachusetts.

Dennett, D. C. (1987). *The Intentional Stance*. MIT Press.

Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*. Wiley, New York.

Fasulo, D. (1999). An analysis of recent work on clustering algorithms. Technical report, University of Washington.

Ferrell, C. (1998). Learning by scaffolding. MIT Ph.D. Thesis Proposal.

Ferrell, C. and Kemp, C. (1996). An ontogenetic perspective to scaling sensorimotor intelligence. In *Embodied Cognition and Action: Papers from the 1996 AAAI Fall Symposium*. AAAI Press.

Firby, R. J. (1994). Task networks for controlling continuous processes. In *Proceedings of the Second International Conference on AI Planning Systems*, Chicago IL.

Fitzpatrick, P. and Metta, G. (2002). Towards manipulation-driven vision. In *IEEE/RSJ Conference on Intelligent Robots and Systems*.

Fogassi, L., Gallese, V., Fadiga, L., Luppino, G., Matelli, M., and Rizzolatti, G. (1996). Coding of peripersonal space in inferior premotor cortex (area F4). *Journal of Neurophysiology*, pages 141–157.

Folsom, T. C. and Pinter, R. B. (1998). Primitive features by steering, quadrature, and scale. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1161–1173.

Freeman, W. (1992). *Steerable filters and the local analysis of image structure*. PhD thesis, Media Arts and Sciences, MIT.

Freeman, W. T. and Adelson, E. H. (1991). The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906.

Gallese, V., Fadiga, L., Fogassi, L., and Rizzolatti, G. (1996). Action recognition in the premotor cortex. *Brain*, 119:593–609.

Gansner, E. R. and North, S. C. (2000). An open graph visualization system and its applications to software engineering. *Software: Practice and Experience*, 30(11):1203–1233.

Garland, A. and Lesh, N. (2001). Learning hierarchical task models by demonstration. Technical report, Mitsubishi Electric Research Laboratories.

Gat, E. (1996). ESL: A language for supporting robust plan execution in embedded autonomous agents. In *Plan Execution: Problems and Issues: Papers from the 1996 AAAI Fall Symposium*, pages 59–64. AAAI Press, Menlo Park, California.

Gibson, J. J. (1977). The theory of affordances. In Shaw, R. and Bransford, J., editors, *Perceiving, acting and knowing: toward an ecological psychology*, pages 67–82. Hillsdale NJ: Lawrence Erlbaum Associates Publishers.

Glass, J., Chang, J., and McCandless, M. (1996). A probabilistic framework for feature-based speech recognition. In *Proceedings of the International Conference on Spoken Language Processing*, pages 2277–2280.

Glass, J. and Weinstein, E. (2001). SPEECHBUILDER: Facilitating spoken dialogue systems development. In *Proceedings of the Seventh European Conference on Speech Communication and Technology*, Aalborg, Denmark.

Goldberg, D. and Mataric, M. J. (1999). Augmented markov models. Technical report, USC Institute for Robotics and Intelligent Systems.

Goldberg, M. E. (2000). The control of gaze. In Kandel, E. R., Schwartz, J. H., and Jessell, T. M., editors, *Principles of Neural Science*. McGraw-Hill, 4rd edition.

Goodman, N. (1983). *Fact, Fiction, and Forecast*. Harvard University Press, Cambridge, Massachusetts.

Gorin, A., Petrovksa-Delacrtaz, D., Riccardi, G., and Wright, J. (1999). Learning spoken language without transcriptions. In *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*, Colorado.

Granlund, G. H. (1978). In search of a general picture processing operator. *Computer Graphics and Image Processing*, 8(2):155–173.

Graziano, M. S. A., Hu, X., and Gross, C. G. (1997). Visuo-spatial properties of ventral premotor cortex. *Journal of Neurophysiology*, 77:2268–2292.

Grice, H. (1974). Logic and conversation. In Cole, P. and Morgan, J., editors, *Syntax and semantics*, volume 3, pages 41–58. Academic Press, New York.

Grimson, W. E. L. and Huttenlocher, D. P. (1990). On the sensitivity of the Hough transform for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(3):255–274.

Haigh, K. Z. and Veloso, M. M. (1998). Planning, execution and learning in a robotic agent. In *Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems*, pages 120 – 127.

Halliday, M. (1975). *Learning How to Mean: Explorations in the Development of Language*. Elsevier, New York, NY.

Harnad, S. (2002). Symbol grounding and the origin of language. In Scheutz, M., editor, *Computationalism: New Directions*, pages 143–158. MIT Press.

Harville, M., Rahimi, A., Darrell, T., Gordon, G., and Woodfill, J. (1999). 3D pose tracking with linear depth and brightness constraints. In *Proceedings of the International Conference on Computer Vision*, pages 206–213.

Hauser, M. D. (1996). *Evolution of Communication*. MIT Press.

Hazen, T. and Bazzi, I. (2001). A comparison and combination of methods for OOV word detection and word confidence scoring. In *Proceedings of the International Conference on Acoustics*, Salt Lake City, Utah.

Heinzmann, J. and Zelinsky, A. (1997). Robust real-time face tracking and gesture recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 2, pages 1525–1530.

Held, R. and Hein, A. (1963). Movement-produced stimulation in the development of visually-guided behavior. *Journal of Comparative and Physiological Psychology*, 56:872–876.

Hirschberg, J., Litman, D., and Swerts, M. (1999). Prosodic cues to recognition errors. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*.

Horn, B. K. P. (1986). *Robot Vision*. MIT Press.

Horprasert, T., Yacoob, Y., and Davis, L. S. (1997). An anthropometric shape model for estimating head orientation. In *Proceedings of the Third International Workshop on Visual Form*, Capri, Italy.

Horswill, I. (1993). *Specialization of Perceptual Processes*. PhD thesis, MIT.

Hu, M. K. (1962). Visual pattern recognition by moment invariants. In *IRE Transactions on Information Theory 8*, pages 179–187.

Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154.

Hueckel, M. H. (1971). An operator which locates edges in digitized pictures. *Journal of the Association for Computing Machinery*, 18(1):113–125.

Hurley, S. L. (2001). Perception and action: Alternative views. *Synthese*, 129:3–40.

Iacoboni, M., Woods, R. P., Brass, M., Bekkering, H., Mazziotta, J. C., and Rizzolatti, G. (1999). Cortical mechanisms of human imitation. *Science*, 286:2526–2528.

Jacobs, D. (1996). Robust and efficient detection of convex groups. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(1):23–37.

Jacobs, D., Rokers, B., , and Rudra, A. (2001). Fragment completion in humans and machines. In *Neural Information Processing Systems*.

Jeannerod, M. (1997). *The Cognitive Neuroscience of Action*. Blackwell Publishers Inc., Cambridge Massachusetts and Oxford UK.

Jeanrenaud, P., Ng, K., Siu, M., Rohlicek, J., and Gish, H. (1993). Phonetic-based word spotter: Various configurations and application to event spotting. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH)*.

Jia, Y.-B. and Erdmann, M. (1998). Observing pose and motion through contact. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Johnson, S. P. (in press). Building knowledge from perception in infancy. In Gershkoff-Stowe, L. and Rakison, D., editors, *Building object categories in developmental time*. Hillsdale, Mahwah, NJ.

Jusczyk, P. (1997). *The Discovery of Spoken Language*. Cambridge: MIT Press.

Jusczyk, P. and Aslin, R. (1995). Infants' detection of the sound patterns of words in fluent speech. *Cognitive Psychology*, 29:1–23.

Kaelbling, L. P., Littman, M. L., and Moore, A. P. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.

Kaelbling, L. P., Oates, T., Hernandez, N., and Finney, S. (2001). Learning in worlds with objects. In *AAAI Spring Symposium*.

Kamp, H. (1981). A theory of truth and semantic representation. In J, G., Janssen, T., and Stokhof, M., editors, *Formal Methods in the Study of Language*, pages 277–322. Mathematical Center Tract 135, Amsterdam.

Kass, M. and Witkin, A. (1987). Analyzing oriented patterns. *Computer Vision, Graphics, and Image Processing*, 37:362–385.

Kirkham, N. Z., Slemmer, J. A., and Johnson, S. P. (2002). Visual statistical learning in infancy: evidence of a domain general learning mechanism. *Cognition*, 83(2):B35–B42.

Kirsh, D. (1995a). Complementary strategies: Why we use our hands when we think. In *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, Hillsdale, NJ. Lawrence Erlbaum.

Kirsh, D. (1995b). The intelligent use of space. *Artificial Intelligence*, 73:31–68.

Kirsh, D. (1996). Adapting the environment instead of oneself. *Adaptive Behavior*, 4(3/4):415–452.

Klarquist, W. and Bovik, A. (1998). FOVEA: A foveated vergent active stereo system for dynamic three-dimensional scene recovery. *IEEE Transactions on Robotics and Automation*, 14(5):755–770.

Klingspor, V., Demiris, J., and Kaiser, M. (1997). Human-robot-communication and machine learning. *Applied Artificial Intelligence Journal*, 11:719–746.

Koga, Y., Kondo, K., Kuffner, J., and Latombe, J.-C. (1994). Planning motions with intentions. *Computer Graphics*, 28:395–408.

Konishi, S. M., Yuille, A. L., Coughlan, J. M., and Zhu, S. C. (2003). Statistical edge detection: Learning and evaluating edge cues. *Pattern Analysis and Machine Intelligence*, 25(1):57–74.

Kozima, H. (1998). Attention-sharing and behavior-sharing in human-robot communication. In *IEEE International Workshop on Robot and Human Communication (ROMAN-98)*, pages 9–14.

Kozima, H., Nakagawa, C., and Yano, H. (2002). Emergence of imitation mediated by objects. In *The Second International Workshop on Epigenetic Robotics*, Edinburgh, Scotland.

Kubota, T. (1995). *Orientational filters for real-time computer vision problems*. PhD thesis, School of Electrical and Computer Engineering, Georgia Institute of Technology.

Kubota, T. and Alford, C. O. (1995). Computation of orientational filters for real-time computer vision problems i: Implementation and methodology. *Real-Time Imaging*, 1(4):261–281.

La Cascia, M., Sclaroff, S., and Athitsos, V. (2000). Fast, reliable head tracking under varying illumination: An approach based on robust registration of texture-mapped 3D models. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 22(4), pages 322–336.

Lakoff, G. (1987). *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. University of Chicago Press, Chicago, Illinois.

Lamiroy, B. and Gros, P. (1996). Rapid object indexing and recognition using enhanced geometric hashing. In *Proceedings of the 4th European Conference on Computer Vision*, volume 1, pages 59–70, Cambridge, England.

Lenat, D. B. (1995). CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

Marjanovic, M. (1995). Learning functional maps between sensorimotor systems on a humanoid robot. Master's thesis, MIT Department of Electrical Engineering and Computer Science.

Marjanović, M. J. (2003). *Teaching an Old Robot New Tricks: Learning Novel Tasks via Interaction with People and Things*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA.

Marjanović, M. J., Scassellati, B., and Williamson, M. M. (1996). Self-taught visually-guided pointing for a humanoid robot. In *From Animals to Animats: Proceedings of 1996 Society of Adaptive Behavior*, pages 35–44, Cape Cod, Massachusetts.

Markman, E. M. (1989). *Categorization and naming in children: problems of induction*. MIT Press, Cambridge, Massachusetts.

Martin, D., Fowlkes, C., and Malik, J. (2002). Learning to detect natural image boundaries using brightness and texture. In *Sixteenth Annual Conference on Neural Information Processing Systems*.

Mataric, M. J. (1990). A distributed model for mobile robot environment-learning and navigation. Technical Report AITR-1228, Massachusetts Institute of Technology.

Mataric, M. J. (2000). Getting humanoids to move and imitate. *IEEE Intelligent Systems*, 15(4):18–24.

Matsusaka, Y. and Kobayashi, T. (1999). Human interface of humanoid robot realizing group communication in real space. In *Proceedings of the Second International Symposium on Humanoid Robots*, pages 188–193.

McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B. and Michie, D., editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press.

McDermott, J. (2000). Experiments on junctions in real images. Master's thesis, University College London.

McGrenere, J. and Ho, W. (2000). Affordances: Clarifying and evolving a concept. In *Proceedings of Graphics Interface*.

McKenna, S. and Gong, S. (1998). Real-time face pose estimation. *International Journal on Real Time Imaging, Special Issue on Real-time Visual Monitoring and Inspection.*, 4:333–347.

Mester, R. (2000). Orientation estimation: conventional techniques and a new non-differential method. In *European Signal Processing Conference*, Tampere, Finland.

Metta, G. (2000). *Babybot: a study into sensorimotor development*. PhD thesis, LIRA-Lab, DIST.

Metta, G., Sandini, G., and Konczak, J. (1999). A developmental approach to visually-guided reaching in artificial systems. *Neural Networks*, 12:1413–1427.

Minsky, M. (1985). *The Society of Mind*. Simon and Schuster, New York.

Minsky, M. (1990). Logical vs. analogical or symbolic vs. connectionist or neat vs. scruffy. In Winston, P. H., editor, *Artificial Intelligence at MIT, Expanding Frontiers*, volume 1. MIT Press. Reprinted in AI Magazine, 1991.

Mitchell, T. (1980). The need for biases in learning generalizations. Technical report, Computer Science Department, Rutgers University.

Moll, M. and Erdmann, M. A. (2001). Reconstructing shape from motion using tactile sensors. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, HI.

Mou, X. and Zue, V. (2001). Sub-lexical modelling using a finite state transducer framework. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Salt Lake City, Utah.

Münch, S., Kreuziger, J., Kaiser, M., and Dillmann, R. (1994). Robot programming by demonstration (RPD) – using machine learning and user interaction methods for the development of easy and comfortable robot programming systems. In *Proceedings of the 24th International Symposium on Industrial Robots*.

Murata, A., Gallese, V., Luppino, G., Kaseda, M., and Sakata, H. (2000). Selectivity for the shape, size, and orientation of objects for grasping in neurons of monkey parietal area AIP. *Journal of Neurophysiology*, 83(5):2580–2601.

Murphy, K. P. (1996). Passively learning finite automata. Technical report, Santa Fe Institute.

Nayar, S. K., Nene, S. A., and Murase, H. (1996). Real-time 100 object recognition system. In *Proceedings of IEEE International Conference on Robotics and Automation*.

Needham, A. (2001). Object recognition and object segregation in 4.5-month-old infants. *Journal of Experimental Child Psychology*, 78(1):3–22.

Needham, A. and Baillargeon, R. (1997). Object segregation in 8-month-old infants. *Cognition*, 62(2):121–159.

Nelson, R. C. and Selinger, A. (1998). A cubist approach to object recognition. In *Proceedings of the International Conference on Computer Vision*, pages 614–621.

Nicolescu, M. and Mataric, M. J. (2001). Experience-based learning of task representations from human-robot interaction. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 463–468, Alberta, Canada.

Nothdurft, H. C. (1993). The role of features in preattentive vision: Comparison of orientation, motion and color cues. *Vision Research*, 33:1937–1958.

Oates, T. (1999). Identifying distinctive subsequences in multivariate time series by clustering. In *Knowledge Discovery and Data Mining*, pages 322–326.

Oates, T., Eyler-Walker, Z., and Cohen, P. (2000). Toward natural language interfaces for robotic agents: Grounding linguistic meaning in sensors. In *Proceedings of the 4th International Conference on Autonomous Agents*, pages 227–228.

Oates, T., Jensen, D., and Cohen, P. (1998). Discovering rules for clustering and predicting asynchronous events. In *Predicting the Future: AI Approaches to Time-Series Problems*, pages 73–79. AAAI Press.

Oztop, E. (2002). *Modeling the Mirror: Grasp Learning and Action Recognition*. PhD thesis, University of Southern California.

Oztop, E. and Arbib, M. A. (2002). Schema design and implementation of the grasp-related mirror neuron system. *Biological Cybernetics*, 87(2):116–140.

Papert, S. (1966). The summer vision project. Memo AIM-100, MIT AI Lab.

Paulos, E. (1999). Fast construction of near optimal probing strategies. Master's thesis, University of California, Berkeley.

Pelz, J. B. (1995). *Visual Representations in a Natural Visuo-motor Task*. PhD thesis, Brain and Cognitive Science, University of Rochester.

Pepperberg, I. (1990). Referential mapping: A technique for attaching functional significance to the innovative utterances of an african grey parrot. *Applied Psycholinguistics*, 11:23–44.

Perona, P. and Malik, J. (1990). Detecting and localizing edges composed of steps, peaks and roofs. In *Proceedings of the Third International Conference of Computer Vision*, pages 52–57, Osaka, Japan.

Pilu, M., Fitzgibbon, A., and Fisher, R. (1996). Ellipse-specific direct least-square fitting. In *IEEE International Conference on Image Processing*, Lausanne.

Pratchett, T. (1986). *The Light Fantastic*. St. Martin's.

Pratchett, T. (1989). *Diggers*. Transworld Publishers.

Pratchett, T. (1990). *Moving Pictures*. Gollancz.

Pratchett, T. (1991a). *Reaper Man*. Gollancz.

Pratchett, T. (1991b). *Witches Abroad*. Gollancz.

Pratchett, T. (1992a). *Only You Can Save Mankind*. Doubleday.

Pratchett, T. (1992b). *Small Gods*. Gollancz.

Pratchett, T. (1996). *Hogfather*. Gollancz.

Pratchett, T. (1999). *The Fifth Elephant*. Doubleday.

Quine, W. V. O. (1960). *Word and object*. Harvard University Press, Cambridge, Massachusetts.

Rahimi, A., Morency, L.-P., and Darrell, T. (2001). Reducing drift in parametric motion tracking. In *roceedings of the International Conference on Computer Vision*.

Rizzolatti, G. and Arbib, M. A. (1998). Language within our grasp. *Trends in Neurosciences*, 21:188–194.

Ross, M. G. (2000). Exploiting texture-motion duality in optical flow and image segmentation. Master's thesis, Massachusetts Institute of Technology.

Roy, D. (1999). *Learning Words from Sights and Sounds: A Computational Model*. PhD thesis, MIT.

Roy, D., Gorniak, P., Mukherjee, N., and Juster, J. (2002). A trainable spoken language understanding system for visual object selection. In *Proceedings of the International Conference of Spoken Language Processing*.

Roy, D., Hsiao, K., and Mavridis, N. (2003). Conversational robots: Building blocks for grounding word meanings. submitted to HLT-NAACL03 Workshop on Learning Word Meaning from Non-Linguistic Data.

Roy, D. and Pentland, A. (2002). Learning words from sights and sounds: A computational model. *Cognitive Science*, 26(1):113–146.

Sandini, G., Gandolfo, F., Grosso, E., and Tistarelli, M. (1993). Vision during action. In Aloimonos, Y., editor, *Active Perception*, pages 151–190. Lawrence Erlbaum Associates, Hillsdale, NJ.

Scassellati, B. (1998). A binocular, foveated active vision system. Technical Report 1628, MIT Artificial Intelligence Lab Memo.

Scassellati, B. (1999). Imitation and mechanisms of joint attention: A developmental structure for building social skills on a humanoid robot. In Nehaniv, C. L., editor, *Computation for Metaphors, Analogy and Agents*, volume 1562 of *Springer Lecture Notes in Artificial Intelligence*. Springer-Verlag.

Scassellati, B. (2000). Theory of mind for a humanoid robot. In *Proceedings of the First International IEEE/RSJ Conference on Humanoid Robotics*.

Scassellati, B. (2001). *Foundations for a Theory of Mind for a Humanoid Robot*. PhD thesis, MIT Department of Electrical Engineering and Computer Science.

Schneiderman, H. and Kanade, T. (2002). Object detection using the statistics of parts. *International Journal of Computer Vision*.

Schwartz, E. L., Greve, D. N., and Bonmassar, G. (1995). Space-variant active vision: Definition, overview and examples. *Neural Networks*, 8(7-8):1297–1308.

Selinger, A. (2001). *Analysis and Applications of Feature-Based Object Recognition*. PhD thesis, University of Rochester, Rochester, New York.

Sherrah, J. and Gong, S. (2000). Fusion of perceptual cues for robust tracking of head pose and position. In *Pattern Recognition*, volume 34.

Shi, J. and Tomasi, C. (1994). Good features to track. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593 – 600.

Sigal, L., Sclaroff, S., and Athitsos, V. (2000). Estimation and prediction of evolving color distributions for skin segmentation under varying illumination. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 152–159.

Sinha, P. (1994). Object recognition via image invariants: A case study. *Investigative Ophthalmology and Visual Science*, 35:1735–1740.

Smith-Mickelson, J. (2000). Design and application of a head detection and tracking system. Master's thesis, MIT.

Souvignier, B., Kellner, A., Rueber, B., Schramm, H., and Seide, F. (2000). The thoughtful elephant: strategies for spoken dialog systems. *IEEE Transactions on Speech and Audio Processing*, 8(1):51–62.

Sperry, R. W. (1952). Neurology and the mind-brain problem. *American Scientist*, 40:291–312.

Stauffer, C. (1999). Adaptive background mixture models for real-time tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 246–252.

Stauffer, C. and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Fort Colins, CO.

Steels, L. (1996). Emergent adaptive lexicons. In *Proceedings of the fourth international conference on simulation of adaptive behavior*, pages 562–567, Cape Cod, MA.

Strom, J., Jebara, T., Basu, S., and Pentland, A. (1999). Real time tracking and modeling of faces: An EKF-based analysis by synthesis approach. In *Proceedings of the Modelling People Workshop, in the IEEE International Conference on Computer Vision*.

Swain, M. J. and Ballard, D. H. (1991). Color indexing. *International Journal of Computer Vision*, 7(1).

Swerts, M., Hirschberg, J., and Litman, D. (2000). Corrections in spoken dialogue systems. In *Proceedings of the International Conference on Spoken Language Processing*, Beijing, China.

Tarr, M. and Black, M. (1994). A computational and evolutionary perspective on the role of representation in vision. *CVGIP: Image Understanding*, 60(1):65–73.

Thrun, S. (1998). A framework for programming embedded systems: Initial design and results. Technical Report CMU-CS-98-142, Carnegie Mellon University.

Tomasello, M. (1997). The pragmatics of word learning. *Japanese Journal of Cognitive Science*, 4:59–74.

Trevarthen, C. (1979). Communication and cooperation in early infancy: a description of primary intersubjectivity. In Bullowa, M., editor, *Before Speech: The beginning of interpersonal communication*. Cambridge University Press.

Ullman, S. (1984). Visual routines. *Cognition*, 18:97–159. (Also in: Visual Cognition, S. Pinker ed., 1985).

Vaidya, P. (1988). Geometry helps in matching. In *Proceedings of the twentieth annual ACM symposium on theory of computing*, pages 422–425. ACM Press.

Varchavskaia, P., Fitzpatrick, P., and Breazeal, C. (2001). Characterizing and processing robot-directed speech. In *Proceedings of the International IEEE/RSJ Conference on Humanoid Robotics*, Tokyo, Japan.

Viola, P. and Jones, M. (2001). Robust real-time object detection. Technical report, COMPAQ Cambridge Research Laboratory, Cambridge, MA.

Vygotsky, L. (1962). *Thought and language*. MIT Press, Cambridge, MA.

Wang, C. and Brandstein, M. (1998). A hybrid real time face tracking system. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Seattle.

Wang, X. (1996). Planning while learning operators. In Drabble, B., editor, *Proceedings of the Third International Conference on Artificial Intelligence Planning Systems*, pages 229–236. AAAI Press.

Weng, J., Hwang, W. S., Zhang, Y., Yang, C., and Smith, R. (2000a). Developmental humanoids: Humanoids that develop skills automatically. In *Proceedings of the first IEEE-RAS International Conference on Humanoid Robots*, Cambridge, MA.

Weng, J., McClelland, J., Pentland, A., Sporns, O., Stockman, I., Sur, M., and Thelen, E. (2000b). Autonomous mental development by robots and animals. *Science*, 291(5504):599–600.

Werker, J., Lloyd, V., Pegg, J., and Polka, L. (1996). Putting the baby in the bootstraps: Toward a more complete understanding of the role of the input in infant speech processing. In Morgan, J. and Demuth, K., editors, *Signal to Syntax: Bootstrapping From Speech to Grammar in Early Acquisition*, pages 427–447. Lawrence Erlbaum Associates: Mahwah, NJ.

Williamson, M. (1995). Series elastic actuators. Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.

Williamson, M. (1998a). Neural control of rhythmic arm movements. *Neural Networks*, 11(7-8):1379–1394.

Williamson, M. (1999). *Robot Arm Control Exploiting Natural Dynamics*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.

Williamson, M. M. (1996). Postural primitives: Interactive behavior for a humanoid robot arm. In *Fourth International Conference on Simulation of Adaptive Behavior*, pages 124–131, Cape Cod, Massachusetts.

Williamson, M. M. (1998b). Exploiting natural dynamics in robot control. In *Fourteenth European Meeting on Cybernetics and Systems Research (EMCSR '98)*, Vienna, Austria.

Winston, P. (1970). Learning structure descriptions from examples. In *The Psychology of Computer Vision*, pages 157–209. McGraw-Hill, New York.

Wolfe, J. M., Oliva, A., Horowitz, T. S., Butcher, S. J., and Bompas, A. (2002). Segmentation of objects from backgrounds in visual search tasks. *Vision Research*, 42(28):2985–3004.

Wolfson, H. and Rigoutsos, I. (1997). Geometric hashing: an overview. *IEEE Computational Science and Engineering*, 4:10–21.

Wong, D. and Flynn, M. (1992). Fast division using accurate quotient approximations to reduce the number of iterations. *IEEE Transactions on Computers*, 41(8):981–995.

Wren, C., Azarbayejani, A., Darrell, T., and Pentland, P. (1997). Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Y. Wu, K. T. and Huang, T. S. (2000). Wide-range, person- and illumination-insensitive head orientation estimation. In *Proceedings of the International Conference on Face and Gesture Recognition*, Grenoble, France.

Yip, K. and Sussman, G. J. (1997). Sparse representations for fast, one-shot learning. In *Proceedings of the National Conference on Artificial Intelligence*.

Ziou, D. and Tabbone, S. (1997). Edge detection techniques- an overview. Technical report, Dept Math & Informatique. Universit de Sherbrooke, 1997.

Zue, V. and Glass, J. (2000). Conversational interfaces: Advances and challenges. *Proceedings of the IEEE, Special Issue on Spoken Language Processing*, Vol. 88.

Zue, V., Glass, J., Plifroni, J., Pao, C., and Hazen, T. (2000). Jupiter: A telephone-based conversation interface for weather information. *IEEE Transactions on Speech and Audio Processing*, 8:100–112.