ROBOTICS AND PERCEPTION

ROBOTICS AND PERCEPTION

ROBOTICS

Combining Probabilities, Failures and Safety in Robot Control

Alberto Finzi, Fiora Pirri
DIS - Università degli Studi di Roma "La Sapienza"
Via Salaria 113, 00198, Roma, Italy
{ finzi,pirri }@dis.uniroma1.it

Abstract

We present a formal framework for treating both incomplete information in the initial database and possible failures during an agent's execution of a course of actions. These two aspects of uncertainty are formalized by two different notions of probability. We introduce also a concept of expected probability, which is obtained by combining the two previous notions. Expected probability accounts for the probability of a sentence on the hypothesis that the sequence of actions needed to make it true might have failed. Expected probability leads to the possibility of comparing courses of actions and verifying which is more safe.

1 Introduction

Uncertainty, both about the domain in which a robot has to operate and about the effective sequence of actions it is executing, is one of the most crucial problem that cognitive robotics has to face in robot control. In this paper we present a formal framework for dealing with uncertainty both in the initial database and during the execution of a course of actions: actions can fail. The failure of an action can lead to exogenous events, e.g. if the agent fails to pick up a block then the exogenous event that the block falls to the floor can take place. The complete framework is designed for autonomous manipulators endowed with perception. Perception, in fact, can be used to diagnose what really happened during a failed execution (a similar approach is taken in [9]). Here, leaving perception aside, we shall concentrate on the different aspects concerning uncertainty:

1. *Incomplete initial information.* The most prominent logical approaches to probability are [14; 4] and [11; 1; 21; 13; 7; 17]. We introduce in the Situation Calculus [15; 20] a formal theory, which turns out to be equivalent to the possible world approach, under certain restrictions (e.g. [13]). Instead of possible worlds, we propose a suitable axiomatization inducing possible state descriptions [4] therefore relying on a classical semantics. Probability of sentences is entailed by the axiomatization. We do not provide nested probabilities.

2. *Incomplete information for a course of actions, and exogenous events.* A logical approach to failures has been presented in [2]. In general, the problem of failures during the execution of plans is thoroughly treated in the planning community (see e.g. [16; 12]. To cope with the non deterministic effect of actions during the execution of a plan, say, we extend the classical language and axiomatization of the Situation Calculus with stochastic actions and events so that different possible runs, for a given sequence of actions, can be obtained. However we preserve the core ontology of the Situation Calculus in which actions are deterministic and the whole dynamic of change is idealized. In fact, each run of stochastic actions is obtained as an expansion of a fixed sequence of deterministic actions. An expansion is formed by considering both probability of success and failure for each action, and exogenous events. We make a closed world hypothesis about exogenous events: the exogenous events are all and only those fixed in the theory. Probability of events is empirical, it accounts for the relative frequency of properties of events like the usual failures of an agent, due to its normal behavior or the circumstances under which it operates.

The probability of a sentence depends only on the amount of information available in the initial database, i.e. with state descriptions. Probability of actions, on the other hand, depends on properties of events, like a robot behavior. With the notion of expected probability we combine the two former notions (the logical and frequency ones) into a new probability that can account for the probability of a sentence after the execution of a course of actions, in the hypothesis that this sequence can fail in several different ways and exogenous actions can occur. This idea captures in some sense the dynamic execution under incomplete information ([8]). An example is the probability that the screwdriver is on the floor, given that the action the robot has to execute is to pick it up, and its goal is holding it in its hand. Expected probability, combining the two previous probabilities, gives a measure of safety of a sequence of actions chosen to achieve a goal. Therefore it can be used to determine whether a course of actions is more safe than another one. The notion of safety can be suitably extended to treat decision theory in the Situation Calculus (see e.g. [19; 3]).

2 Preliminaries

We consider the Situation Calculus (SC) ([15; 20]) as the core of our logic and language, and we suitably extend it to include new sorts, new symbols for its alphabet and new axioms. We refer the interested reader to [18] for a full presentation of the core logic and language and [9] for more details related to the extensions concerning probability on events. The core consists of the three sorts: *action, situation* and $object = worldEntities \cup genericObjects$. Here the sort $genericObject$ is a catch-all that can include any possible additional sorts needed in an application, like the reals or sets. We have extended the core with three new

sorts: *outcome, stochasticAction, event*. The sort event consists of two sub-sorts, $event = condEvent \cup seqEvent$. Observe that the sorted domain D could be infinite or uncountable even if some of its subsets is finite. For example we are interested in imposing a finite domain for world entities, while we would require probabilities to range over the reals. We shall use the following alphabet, with subscripts and superscripts, to denote terms: a, for variables of sort action; s, for variables of sort situation; sta, for variables of sort stochastic action; e, for variables of sort event; w for variables of the sub-sort $seqEvent$ and u for variables of the sub-sort $condEvent$; p, for variables of sort object denoting probabilities and $x, y, z...$, for variables of sort object (these might be further distinguished, e.g for terms of sort $worldEntity$). The family of terms over SC and the set of formulae $\mathcal{L}_{SC}$ over SC are inductively defined as usual. In particular a *basic sentence* is one formed only by fluents, e.g. $\exists x \forall y(P(x, S_0) \rightarrow Q(x, y, S_0))$, in which fluents take as argument only elements of sort $worldEntities$ (we). A *sentence uniform in σ* is one in which no variable, free or quantified, of sort situation is mentioned. E.g. $\exists s P(t, s)$ is not uniform and $\exists x(P(t, do(transmit(x), S_0)))$, is uniform in $\sigma = do(transmit(x), S_0)$.

Omitted quantifiers are always universal quantifiers.

3 The Basic Theory of Actions

The notions related to a basic theory of action with its set of foundational axioms and domain description axioms, are presented in detail in [20] and in [18]. These axioms represent an agent's prior knowledge about its domain of application, and its behavior, and are denoted by the following:

$$\mathcal{D}_{S_0} \cup \mathcal{D}_{ssa} \cup \mathcal{D}_{una} \cup \mathcal{D}_{ap} \cup \Sigma.$$

Here, $\mathcal{D}_{S_0}$ is the initial database, describing what properties hold initially about the objects in the domain. $\mathcal{D}_{ssa} \cup \mathcal{D}_{ap}$ specify, respectively, the successor state axioms and the action precondition axioms. Σ is the set of foundational axioms for situations, and $\mathcal{D}_{una}$ ensures the uniqueness of each action. We present in the following a simple example to illustrate the basic ontology.

3.1 The basic ontology: a dispatching domain

An agent has to dispatch a string of bits from a source to a receiver and the information it has concerns some of the bits and the constrains that the string has to satisfy. The transmission can fail and thus an exogenous action that $distorts$ the signal can take place. (A robotic oriented example is given in [9]).

Fluents $Source(x, n, s)$: At the source there is a bit $x \in \{0, 1\}$ in position n in situation s. E.g. $Source(1, 1, S_0) \wedge Source(0, 2, S_0)$ specifies that the string 10 is at the source in the initial situation S_0.
$Receiver(x, n, s)$: At the receiver there is a bit $x \in \{0, 1\}$ in position n in situation s.

Control and Exogenous Actions $transmit(n)$. This is a control action with the following effect: the n-th bit at the source is transmitted.
$distort(n)$. This is an exogenous action with the following effect: the complement of the n-th bit at the source is transmitted.

The initial database $\mathcal{D}_{S_0}$ There are properties of the domain that the initial database has to represent. We list some of them, where $\overline{x}$ is complement of x:

1. $\neg Receiver(x, n, S_0)$
2. $Source(x, n, S_0) \rightarrow \neg Source(\overline{x}, n, S_0)$
3. $\exists n.(\forall n'.n' \leq n \rightarrow \exists y Source(y, n', S_0)) \wedge$
   $(\forall x \forall n''.n'' > n \rightarrow \neg Source(x, n, S_0))$

The first axiom says that no string of bits is at the $Receiver$ in S_0. The second says that at the source there is always one fixed bit in position n, the last specifies that the length of the string at the source is a priori fixed. Here for example we can fix $\forall x \neg Source(x, 4, S_0)$. The following constrains to the strings in $Source$ are also in $\mathcal{D}_{S_0}$.
C_1. If a bit b is in position $n > 2$, $\overline{b}$ is either in position $n - 1$ or in position $n - 2$:

$$Source(x, n, S_0) \wedge n > 2 \rightarrow$$
$$Source(\overline{x}, n - 1, S_0) \vee Source(\overline{x}, n - 2, S_0)$$

C_2. If a bit b is in position $n = 2$, $\overline{b}$ is in position $n = 1$:

$$Source(x, n, S_0) \wedge n = 2 \rightarrow Source(\overline{x}, n - 1, S_0)$$

For example: 011 is a legal string. 001 and 111 are not. Finally some axioms will state that bits, which we denote by b are limited to 1 and 0, and positions, which we denote by n, constitute a finite set of natural number. Both the sets are ordered and pairwise disjoint.

Successor State Axioms The successor state axiom for $Receiver$ is:

$$Receiver(x, n, do(a, s)) \equiv$$
$$(a = transmit(n) \wedge Source(x, n, S_0)) \vee$$
$$(a = distort(n) \wedge Source(\overline{x}, n, S_0))$$

4 Incomplete initial information

If the initial database $\mathcal{D}_{S_0}$ is incomplete (see [20; 10]), we can assign a probability to those statements ψ, s.t. $\mathcal{D}_{S_0} \not\models \psi$. So, for example, if a goal G is given, we might be interested in finding a sequence σ of actions, such that $Prob(G(\sigma)) \geq 0.7$ is entailed by the theory. A classical, and certainly the most intuitive approach, is based on a possible world semantics, in which all worlds are assigned the same probabilities (see [2; 13] and the probability of a sentence ψ is obtained by summing up the probabilities of the worlds in which ψ is true. We take an analogous method but we avoid introducing possible worlds. In some sense our approach can be viewed as a way to treat worlds in the object language, under certain restrictions. The basic idea develops Carnap's [4] idea of defining state descriptions. In fact, we construct a set of sentences such that each sentence captures exactly one class of structures of the language. Here a class of structures is obtained by filtration: if two structures are isomorphic, w.r.t. the domain of sort we, then they belong to the same class. If the domain of sort we is finite, then we shall get a finite number of classes. As each state description identifies exactly a unique class of structures, we use these sentences to count those classes of structures which are models of $\mathcal{D}_{S_0}$. Having the number of classes of models of $\mathcal{D}_{S_0}$ we can finally assign probability to sentences, analogously as is done in the possible world approach. The advantage of our approach relies only on the fact that we filter the isomorphic models and we identify them directly in the language by the state description that we add to

the initial database. Observe that our approach is, however, possible just in case the domain of sort we has a fixed finite cardinality.

We shall now give few details about the formalization. Let $\mathcal{D}_{S_0}$ be the initial database, which we assume is a finite set of basic sentences, uniform in S_0. The set of basic sentences uniform in S_0 is denoted, for short, by $\mathcal{L}^{we}$.

Given $\mathcal{D}_{S_0}$ we shall consider a set of *State Description* axioms (SD). SD is defined by the following set:

$$\{P_i^\star = \exists x_1 \cdots \exists x_n \Phi_i(S_0) \wedge (x_1 \neq x_2 \wedge \cdots \wedge\ x_{n-1} \neq x_n) \wedge \forall y(y = x_1 \vee \cdots \vee y = x_n)\}_{i \in I}$$
and
$$P_1^\star \vee \ldots \vee P_q^\star \tag{1}$$

Here n is the cardinality of the domain of sort we, and $q \in I$ is an index which will be specified below (3). $\Phi_i(S_0)$, that appears in (1) is constructed as follows.

Definition 1 *Let X be a set of variables. A function f_X is a* variable renaming isomorphism *iff f_X maps X onto X and is one-to-one. Let A and B be two sets of m-tuple of variables from X.*

$$A =_{f_X} B \text{ iff for each } \langle x_1, \ldots, x_m\rangle \in A \quad \langle f_X(x_1), \ldots, f_X(x_m)\rangle \in B$$

f_X induces an equivalence relation between sets of m-tuples. An equivalence class of m-tuples is:

$$[A]_{f_X} = \{B \mid B =_{f_X} A\}$$

Example 1 *Let $X = (x_1, x_2, x_3)$, $A = \{\langle x_1, x_1\rangle, \langle x_2, x_1\rangle\}$ and $B = \{\langle x_3, x_3\rangle, \langle x_2, x_3\rangle\}$. $f_X(x_1) = x_3$ and $f_X(x_2) = x_2$ is a variable renaming isomorphism, and $A =_{f_X} B$.*

Let ϕ be a basic atom uniform in S_0, of the form $F(\vec{x}, y)$ or $t(\vec{x}, y) = t'$, with $\vec{x}$ possibly empty, $y = S_0$ if it appears, and t, t' terms of sort we. Let $m \in \{\, a \mid \phi^a \in \mathcal{L}^{we}, a$ the arity of $\phi\}$. Let $X = \{x_1, \ldots, x_n\}$:

$$C_m = \{\langle x_1, \ldots, x_m\rangle_j \mid 1 \leq j \leq n^m, x_i \in X\}$$
$\{A_m\}_{i \in I}$ a family of sets of m-tuples, s.t. for each $[B_m]$, $B_m \subset C_m$, only one representative is in $\{A_m\}_{i \in I}$

$$h : \langle i, m\rangle \mapsto \{A_m\}_{i \in I} \quad h, \text{ is bijective and such that } h(i, m) \in \{A_m\}_{i \in I}$$

The above definitions are used to form all the possible m-tuples out of n variables and ensure to gather only one representative for each equivalence class, under the renaming isomorphism. Let us order all the atoms ϕs, as defined above. Let ϕ_k be the k-th element in such ordering Ord, with $|Ord| = \alpha$. $\Phi_i(S_0)$ is defined as follows:

$$\Phi_i(S_0) = \textstyle\bigwedge_{1 \leq k \leq \alpha}(\bigwedge_{j \in h(i,m)} \phi_k^m(\langle x_1, \cdots\ x_m\rangle_j) \wedge \bigwedge_{j \in C_m \setminus h(i,m)} \neg\phi_k^m(\langle x_1, \cdots\ x_m\rangle_j)) \tag{2}$$

Observe that, given $n > 0$ the number of variables, and $m > 0$ the arity of the tuples, and $\phi_k \in Ord$,

$$\lambda_{\phi_k} = 2^{n^m} - \sharp f_X \text{ and } q = \prod_{\phi_k \in Ord} \lambda_{\phi_k} \tag{3}$$

Here $\sharp f_X$ is the number of renaming isomorphisms and $I = \{1, \ldots, \lambda_{\phi_k}\}$.

To show that each $\Phi_i(S_0)$ and, consequently each $P_i^\star$, is uniquely defined, many lemmata have to be used. Moreover an equivalence relation, actually an isomorphism on structures, based only on the domain of sort we, and mirroring the renaming isomorphism, has to be defined. We skip all these details and leave only the main theorem.

Theorem 1 *Let $\mathcal{M}$ be a structure of $\mathcal{L}_{SC}$ with domain of sort we with cardinality n. There exists a unique $P_i^\star$ such that $\mathcal{M} \models P_i^\star$.*

The above theorem tells us that each $P^\star$ is a possible state description and, by the uniqueness, there is a one-to-one correspondence between the $P^\star$s and the classes of structures of $\mathcal{L}_{SC}$, with domain of sort we of fixed cardinality n.

Theorem 2 *Let ϕ be a basic sentence, uniform in S_0, such that $SD \cup \{\phi\}$ is consistent. There is an ordering on the $P^\star$s, such that:*

$$SD \models \phi \equiv (P_1^\star \vee \ldots \vee P_k^\star) \wedge \bigwedge_{k<j\leq q} \neg P_j^\star$$

To count probabilities we need to extend $\mathcal{L}_{SC}$ with a number $p_1, \ldots, p_q$ of parameters.

Lemma 1 *Let $p_1, \ldots, p_q$ be new parameters of the language, and let us add the following to $\mathcal{D}_{S_0}$:*

$$(\textstyle\bigwedge_i (p_i = 1 \equiv P_i^\star) \wedge (p_i = 1 \equiv p_i \neq 0) \tag{4}$$

Then there is an ordering on the $P^\star$s such that $\mathcal{D}_{S_0} \cup SD$ entails both:

1. $\wedge\mathcal{D}_{S_0} \equiv P_1^\star \vee \ldots \vee P_j^\star \wedge \bigwedge_{j<k\leq n} \neg P_k^\star$
2. $\bigvee_{1\leq i\leq j} p_i = 1 \wedge \bigwedge_{j<k\leq n} p_k = 0$

Observe that the role of (4) is to assign to each model $\mathcal{M}$ of $\mathcal{D}_{S_0}$, which is also a model of a unique $P_i^\star$ a value 1 for the associated p_i. On the other if $\mathcal{M}_i$ is not a model of $\mathcal{D}_{S_0}$ the p_i associated with its unique state description $P_i^\star$, will have everywhere value 0. We can, now, count the models of $\mathcal{D}_{S_0}$ as follows, where $\sharp\mathcal{M}$ is a new parameter:

$$\sharp\mathcal{M} = p_1 + \ldots + p_q \equiv \bigvee_{1\leq i\leq j} p_i = 1 \wedge \bigwedge_{j<k\leq q} p_k = 0 \tag{5}$$

We are, now, ready to introduce the probability of sentences based on the equivalence classes of models of $\mathcal{D}_{S_0} \cup SD$. This can be done at the object level just by operating on the $P^\star$'s. First we define:

Definition 2 *Let $\mathcal{D}_{S_0}$ be equivalent, for some ordering on the $P^\star$s, to $\bigvee_{1\leq i\leq j} P_i^\star \wedge \bigwedge_{j<k\leq n}^{i\neq k} \neg P_k^\star$, let $r \in \mathbb{R}$:*

$$Prob(P_i^\star) = \frac{1}{r} \text{ iff } r = \sharp\mathcal{M} \wedge P_1^\star \vee \ldots \vee P_j^\star \wedge \bigwedge_{j<k\leq n}^{i\neq k} \neg P_k^\star$$
$$Prob(P_i^\star) = 0 \text{ iff } \neg P_i^\star$$

It follows, by Lemma 1 and Definition 5, that $\sum_i Prob(P_i^\star) = 1$.

The following definition will give us the probability of a sentence with respect to the state descriptions.

Definition 3 *For any basic formula ϕ, uniform in S_0:*

$$Prob(\phi) = \sum_{1\leq i\leq n} Prob(P_i^\star) \text{ iff } \phi \equiv \bigvee_{1\leq i\leq j} P_i^\star \wedge \bigwedge_{j<k\leq n} \neg P_k^\star \tag{6}$$

Theorem 3 *The following properties are satisfied for each basic sentence ϕ, either uniform in S_0, or equivalent to a sentence uniform in S_0, in all models of $\mathcal{D}_{S_0} \cup SD$:*

1. $0 \leq Prob(\phi)$
2. $Prob(\top) = 1$
3. $Prob(\phi) = Prob(\phi \wedge \psi) + Prob(\phi \wedge \neg\psi)$
4. *If* $\mathcal{D}_{S_0} \cup SD \models \phi \equiv \psi$ *then* $Prob(\phi) = Prob(\psi)$

It follows that, if ψ is uniform in S_0 then:

$$\mathcal{D}_{S_0} \cup SD \models \psi \text{ iff } Prob(\psi) = 1.$$

Observe that differently from Bacchus and Halpern we do not allow any nesting of probabilities. In other words we do not admit sentences of the form: $Prob(Prob(\phi) \geq 0.3)$. However, given the above limitations, and the strict constraint that the domain of sort world entities has a fixed, finite, cardinality, the above theorem shows that our logic is equivalent to Halpern's logic, when to Halpern's axiomatization the finiteness axiom FIN is added.

Example 2 (Continued) Let $D_{we} = \{\{0,1\},\{1,2,3\}\}$. We assume that no information about the three bits in the source's string in S_0 is given, but the string is three bits and it satisfies C_1 and C_2. The possible strings are:

$$st_1 = 100, st_2 = 101, st_3 = 010, st_4 = 011$$

Each string corresponds to a state description. E.g. 100 is captured by P_1^* with the following Φ_1:

$$\Phi_1(s_0) = \bigwedge_{i\in\{0,1\}} b_i = i \wedge \bigwedge_{1\leq i\leq 3} n_i = i \wedge \bigwedge_{i,j} \neg Receiver(b_i, n_j, S_0)$$
$$\wedge Source(b_1, n_1, S_0) \wedge Source(b_0, n_2, S_0) \wedge Source(b_0, n_3, S_0) \wedge$$
$$\bigwedge_{\langle i,j\rangle \notin \{\langle 1,1\rangle, \langle 0,2\rangle, \langle 0,3\rangle\}} \neg Source(b_i, n_j, S_0) \wedge \chi_1$$

Here χ_1 states the ordering defined respectively on bits and positions. P_2^*, P_3^* and P_4^* can be defined analogously. $D_{S_0} \equiv P_1^* \vee P_2^* \vee P_3^* \vee P_4^* \wedge \bigwedge_{4<j<q} \neg P_j^*$ where the P_j are all the other state descriptions admitted by the 5-structures of $\mathcal{L}_{we}$. Hence $\sharp\mathcal{M} = 4$ and $Prob(P_i^*) = \frac{1}{4}$. We want to evaluate the probability of the following sentence:

$$\Psi_s(S_0) \equiv Source(x, n, S_0) \wedge Source(\overline{x}, n+1, S_0)$$

Stating that *each bit b is followed by $\overline{b}$*, which is not entailed by $\mathcal{D}_{S_0}$. As $\Psi_s(S_0) \equiv P_2^* \vee P_3^* \wedge \bigwedge_{j\notin\{2,3\}} \neg P_j^*$ then $Prob(\Psi_s(S_0)) = \frac{1}{2}$.

5 A model for failures

The nice behavior of the Situation Calculus as a logic of change, is that once we have suitably defined truth and probabilities in the initial database, everything about uniform sentences in $\sigma > S_0$ (i.e. about the future of the initial database) can be easily computed in $\mathcal{D}_{S_0}$, using regression. Regression [18] is a proof method that, given a sentence uniform in a situation σ reduces such a sentence to an equivalent sentence in which the only situation mentioned is S_0. Regression amounts to the following equivalence, given a sentence $\psi(\sigma)$:

$$\mathcal{D} \models \forall(\psi(\sigma) \equiv \mathcal{R}(\psi(\sigma)))$$

Here $\mathcal{R}(\psi(\sigma))$ is a sentence uniform in S_0. Now, by the properties of probability shown in Theorem 3, regression allows us to compute the probability for any sentence uniform in σ.

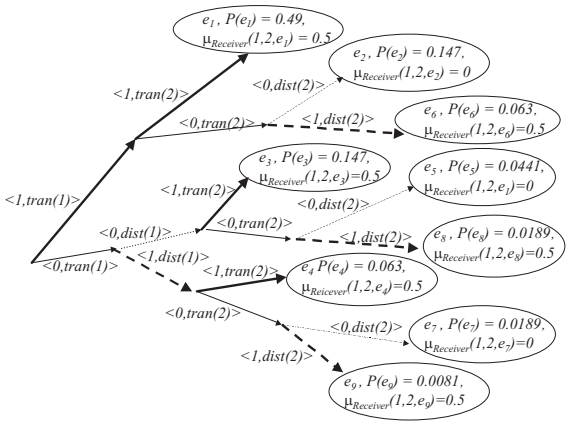


Figure 1: The tree $tree(do(transmit(2), transmit(1), S_0)))$. The dotted arrow means the execution of an exogenous action.

Observe however that we do not have a method to compute the probability of any sentence, what so ever. For example we would not be able to compute the probability of a sentence of the form $\exists s\forall s'\exists s''\exists x\ y(P(x,s) \wedge Q(x,y,s') \rightarrow R(y,x,s''))$. And since the definition of $Prob$ is based on state descriptions, which are closed sentences, we cannot either express sentences of the form: $\forall x Prob(\phi(x))$.

Independently of probabilities, we have so far given an idealized representation of the world in which the effects of actions are exactly those intended, i.e. actions, even exogenous actions, are deterministic. However, due to various circumstances (e.g. noise in the channel), an agent might fail in its intended execution of an action and probabilities on sentences cannot capture this fact. To address this problem, we have extended the specification of an action theory with stochastic actions and events, therefore we introduce a complementary notion of probability as a relative frequency of events. We introduce in the sequel the functions, constants and relations of the extended $\mathcal{L}_{SC}$ that we use to model stochastic actions and probability of events, a more detailed description is in [9].

Stochastic Action. $\langle .,. \rangle : outcome \times action \mapsto stochasticAction$. E.g. $\langle v(a), a\rangle$. With $v : action \mapsto outcome$, e.g. $v(a) = 0$.

A *stochastic action* is a pair $\langle v(a), a\rangle$ with $v(a)$ taking values in the outcome space $outcome = \{0,1\}$, where 0 means that a failed and 1 that it succeeded. Given a sequence of deterministic actions, this sequence expands to a tree of events due to all the possible outcomes of each action. See Figure 1.

To model the tree and the probability distribution we introduce the following notion of event.

Event. $\circ : stochasticAction \times seqEvent \mapsto seqEvent$. $\mid : stochasticAction \times seqEvent \mapsto condEvent$, and the special constant E_0. E.g. $\langle 1, distort(bit2)\rangle \circ \langle 0, transmit(bit1)\rangle \circ E_0$ is a sequence of stochastic actions, i.e. is a term of sort $seqEvent$, hence a term of sort event. An event e is either a sequence event w of sort $seqEvent$ or a conditional event u of sort $condEvent$, according to the following definition:

sequence event :
1. E_0 : the empty event.
2. $(\langle v(a), a\rangle \circ w)$: the event obtained as a result of executing a with value $v(a) \in \{0,1\}$, after the sequence event w.

conditional event :
3. $(\langle v(a), a\rangle \mid w)$: the event obtained as a a result of executing a with value $v(a) \in \{0,1\}$, given the sequence event w

Probability of events. **P**: $stochasticAction \times event \mapsto object$. E.g. $\mathbf{P}(\langle 0, distort(n)\rangle | E_0) = 0.9$. Probability on events, is a functional event fluent **P** which takes as arguments events and returns their probability: **P** maps events into $[0, 1]$.

Finally the following functions and predicates are used for the theory of events.

Measure. $\mu_P : object^n \times seqEvent \mapsto object$. E.g. $\mu_{Source}(bit, pos, E_0) = 0.7$. For each fluent P there is a belief measure μ_P.

$\prec$: $seqEvent \times seqEvent$. E.g. $E_0 \prec \langle v(a), a\rangle \circ E_0$.

Could. $stochasticAction \times event$. E.g. $Could(\langle 1, fall\rangle, S_0)$.

Axioms for probability P We shall not treat here axioms for events, we refer the reader to [9]. These axioms mirror the foundational axioms for situations, and for all events of sort sequence the analogous theorems as for situations [18] are derivable. The properties of **P** are:

1. $\mathbf{P}(e) \geq 0$, for any event e.
2. $\mathbf{P}(E_0) = 1$.
3. $\mathbf{P}(sta \circ e) = \mathbf{P}(sta \mid e) \times \mathbf{P}(e)$
4. $\mathbf{P}(\langle 0, a\rangle | \ e) = 1 - \mathbf{P}(\langle 1, a\rangle \mid e)$.

We define the following function:

$$trans(E_0) = S_0, \ \ trans(\langle 0, a\rangle \circ e) = trans(e)$$
$$trans(\langle 1, a\rangle \circ e) = do(a, trans(e))$$

Let $\nu \in \{0, 1\}$. The following sets can be defined, with $ExAct$ a predicate that sorts out the exogenous actions:

$$tree(s) = \{\langle \nu, a\rangle \circ w \mid ExAct(a) \wedge w \in tree(s) \vee trans(\langle 1, a\rangle \circ w) \sqsubseteq s\}$$
$$leaves(s) = \{w | w \in tree(s) \wedge \neg\exists w'.w' \in tree(s) \wedge w \preceq w'\}$$

Given the above axioms and the above functions, the following properties hold:

1. $\mathbf{P}(sta|e) = \dfrac{\mathbf{P}(sta \circ e)}{\mathbf{P}(e)}$
2. $\mathbf{P}(\langle 1, a\rangle \ |e) + \mathbf{P}(\langle 0, a\rangle \mid e) = 1$
3. Let $leaves(\sigma) = \{w_1, \ldots, w_k\}$ then $\sum_i \mathbf{P}(w_i) = 1$.

5.1 Treating preconditions for stochastic actions

When a stochastic action sta is taken into account we need to know what holds at a given event in order to decide if sta is executable. In fact, we shall use the conditional probabilities to state that, if the preconditions are not satisfied, then the probability of success of the action will be 0. We need, therefore, suitable action preconditions axioms and successor state axioms concerning the dynamic of stochastic actions. Consider the dispatching domain and the control and exogenous actions given in paragraph 3.1: $transmit$ and $distort$. The corresponding stochastic actions are $\langle x, transmit(n)\rangle$ and $\langle x, distort(n)\rangle$, where x is a random variable on the values 1 or 0. Preconditions are defined as follows:

$$Could(\langle x, transmit(n)\rangle, e) \equiv \top$$
$$Could(\langle x, distort(n)\rangle, sta \circ e) \equiv sta = \langle 0, transmit(n)\rangle$$

The conditional probabilities for non exogenous action are:

$$\mathbf{P}(\langle 1, transmit(n)\rangle | e) = p \equiv Could(\langle 1, transmit(n)\rangle, e) \wedge p = 0.7 \vee \neg Could(\langle 1, transmit(n)\rangle, e) \wedge p = 0 \quad (7)$$

The preconditions for the success of an exogenous action like $distort$ depends on the result of a previous action. Therefore the probability of an exogenous actions is conditioned on the *random effect* of stochastic actions.

$$\mathbf{P}(\langle 1, distort(n)\rangle | sta \circ e) = p \equiv Could(\langle 1, distort(n)\rangle, e) \wedge sta = \langle 0, transmit(n)\rangle \wedge p = 0.7 \vee p = 0 \wedge \neg Could(\langle 1, distort(n)\rangle, e) \quad (8)$$

Observe that, by the axioms for **P**, we get a conditional probability for failure, for each of the above stochastic actions:

$$\mathbf{P}(\langle 0, a\rangle \mid e) = 1 - \mathbf{P}(\langle 1, a\rangle \mid e)$$

Successor state axioms for fluent measures. Since in the core language of SC fluents can take as argument only sequences of deterministic actions, by a principle of preservation, we prefer to extend the language with event fluents, namely μ-fluents, which we call *measures of fluents*. For each fluent $F(\vec{t}, s)$ we add a functional event fluent $\mu_F(\vec{t}, e)$, with the same arity as F and such that all arguments which are not of sort situation are the same as F and the argument of sort situation is replaced by an argument of sort event. The value of a μ-fluent is given as follows:

$$\mu_F(\vec{t}, E_0) = Prob(F(\vec{t}, S_0))$$

The successor state axioms for fluent measures capture the dynamic of the domain, in terms of measures, under the circumstances that stochastic actions occurred. These successor state axioms are strictly tied to the successor state axioms for fluents and they have to preserve the following property:

Property 1 *For any μ-fluent $\mu_F(\vec{t}, e)$:*

$$\mu_F(\vec{t}, e) = Prob(\mathcal{R}(F(\vec{t}, trans(e))))$$

According to the above property a simple method can transform a successor state axiom for a fluent into a successor state axiom for the corresponding μ-fluent. In particular, as a successor state axiom has the form $F(\vec{x}, do(a, s)) \equiv \Psi$, then we get $Prob(F(\vec{x}, do(a, s))) = Prob(\Psi)$. By suitable manipulations we get the required successor state axioms.

$$\mu_{Receiver}(x, n, sta \circ e) = p \equiv sta = \langle 1, transmit(n)\rangle \wedge \mu_{Source}(x, n, S_0) = p \vee sta = \langle 1, distort(n)\rangle \wedge p = 1 - \mu_{Source}(x, n, S_0).$$

5.2 Expected Probability and Safety

We want to combine the logical probability on sentences and the event probability defined on stochastic actions to get a third probability (see Theorem 4) that accounts for both incomplete information and possible failures of actions during execution. We are interested to know, given that $G(\sigma)$ is what an agent should achieve, what is the probability that G will hold at σ. We also want to know the probability that a sentence G' will hold at σ, given the evidence that G was to be achieved at σ. To capture these ideas we introduce the notion of *Expected Probability*. Let $Eset(\sigma', \sigma) = \{e_i : e_i \in leaves(\sigma), trans(e_i) = \sigma'\}$. Let $T(\sigma) = \{\sigma' \mid trans(e_i) = \sigma', e_i \in leaves(\sigma)\}$.

Definition 4 (Expected probability in S_0) *Let ϕ and ψ be sentences uniform in σ. The expected probability, in S_0, of ϕ, given evidence ψ is:*

$$EP(\phi(\sigma) \mid \psi(\sigma)) = \sum_{\sigma' \in T(\sigma)} (Prob(\mathcal{R}(\phi(\sigma')) | \mathcal{R}(\psi(\sigma))) \times \sum_{e_i \in Eset(\sigma', \sigma)} \mathbf{P}(e_i))$$

Theorem 4 *EP satisfies the properties of Theorem 3.*

The above defined expected probability allows us to compare two sequences of actions and decide which one is safer:

Definition 5 (Safety) *Let $\phi(\sigma_1)$ and $\phi(\sigma_2)$ be two sentence uniform, respectively, in σ_1 and σ_2. Let $\preceq_{safe}$ be a ordering relation:*

$$\phi(\sigma_1) \preceq_{safe} \phi(\sigma_2) \textit{ iff } EP(\phi(\sigma_1)) \leq EP(\phi(\sigma_2)$$

Example 3 (Continued) Let $\Psi_r(\sigma)$ be the sentence $(Receiver(x, n, \sigma) \wedge Receiver(\vec{x}, n + 1, \sigma))$ where $\sigma = do(transmit(2), do(transmit(1), S_0))$. We consider the stochastic tree expansion of σ. Considering the set $\{\sigma' : \sigma' = trans(e), e \in leaves(\sigma)\}$, we get the following possible executions of σ, see Figure 1.

$\sigma_1 = do(transmit(2), do(transmit(1), S_0))$
$\sigma_2 = do(transmit(1), S_0),\ \sigma_3 = do(transmit(2), S_0)$
$\sigma_4 = do(transmit(2), do(distort(1), S_0)),\ \sigma_5 = S_0$
$\sigma_6 = do(distort(2), do(transmit(1), S_0))$
$\sigma_7 = do(distort(1), S_0),\ \sigma_8 = do(distort(2), S_0)$
$\sigma_9 = do(distort(2), do(distort(1), S_0))$

By regression, $Prob(\mathcal{R}[\Psi_r(\sigma_i)]) = 1/2$, for $i \in \{1, 4, 6, 9\}$ and $Prob(\mathcal{R}[\Psi_r(\sigma_i)]) = 1$ for $i = \{2, 3, 5, 7, 8\}$. Considering the probability of events, from (7) and (8) we get:

$\mathbf{P}(\langle 0, distort(x)\rangle | E_0) = 0.7 \quad \mathbf{P}(\langle 0, transmit(x)\rangle | E_0) = 0.3$
$\mathbf{P}(\langle 1, distort(x)\rangle | E_0) = 0.3 \quad \mathbf{P}(\langle 1, transmit(x)\rangle | E_0) = 0.7$

$\mathbf{P}(e_1) = 0.49 \quad \mathbf{P}(e_2) = 0.147 \quad \mathbf{P}(e_3) = 0.147$
$\mathbf{P}(e_4) = 0.063 \quad \mathbf{P}(e_5) = 0.0441 \quad \mathbf{P}(e_6) = 0.063$
$\mathbf{P}(e_7) = 0.0189 \quad \mathbf{P}(e_8) = 0.0189 \quad \mathbf{P}(e_9) = 0.0081$

By Definition 4 we get $EP(\Psi_r(\sigma)) = 0.68795$.

6 Discussion

The notion of probability that we are concerned with, comes from the tradition of the *logical interpretations* of probability ([21]). The basic idea is that when assertions are not entailed it is always possible to conclude something about these assertions, a logical proximity as a *degree of deducibility*, treated as a probability on statements. The idea of attributing probabilities to logical statements was earlier introduced by Keynes's and illustrated in [14]. Carnap, in [4] addresses a distinction between *degree of confirmation* and *relative frequency*. In [13] Halpern proposes both a logic for statistical measure statements and probability statements. The former provides a probabilistic measure of events defined on a generic object: "the probability that a bird flies is 0.5". The latter is about probabilities of logical assertions: "the probability that Tweety flies is..". Following the tradition of *probability logic*, we have considered the Carnaps'notion of degree of confirmation to declare a probability $Prob$ on sentences and the relative frequency idea to declare probability $\mathbf{P}$ on actions. Bacchus and Halpern in [1; 13] presents two systems for both the statistical and confirmative approaches. A unified theory is presented in [7]. In particular, in [13; 1] axioms and rules of inference are presented and the logic is proved sound and complete (under given restrictions), w.r.t. a semantics based on possible worlds. Differently from Halpern's and Bacchus one, our theory is based on a classical Tarskian semantics. By suitably axiomatizing the initial database we get a semantical definition of probabilities analogous to the possible worlds approach. We introduce axioms for state description, typical *extension formulas* used in finite model theory [5; 6]. Finally we have introduced a third probability, the *expected probability*, which can be used to compute the probability of a sentence given both incomplete knowledge and failures during the execution of a course of actions.

References

[1] F. Bacchus. *Representing and Reasoning with Probabilistic Knowledge*. MIT Press, Cambridge, Mass.,, 1990.

[2] F. Bacchus, J. Halpern, and H. Levesque. Reasoning about noisy sensors in the situation calculus. *Artificial Intelligence*, 111:171–208, 1999.

[3] C. Boutilier, R. Reiter, M. Soutchanski, and S. Thrun. Decision-theoretic, high level agent programming in the situation calculus. In *AAAI-2000*, pages 454–460. AAAI Press, 1999.

[4] R. Carnap. *Logical Foundations of Probability*. University of Chicago Press, Chicago, 1950.

[5] R. Fagin. Probabilities on finite models. *Journal of Symbolic Logic*, 41(1):50–58, 1976.

[6] R. Fagin. Finite-model theory: a personal perspective. *Theoretical Computer Science*, 116:3–31, 1993.

[7] R. Fagin and J. Halpern. Reasoning about knowledge and probability. *ACM*, 41(2):340–367, 1994.

[8] Y. A. Feldman and D. Harel. A probabilistic dynamic logic. *Journal of Computer and System Sciences*, 28:193–215, 1984.

[9] A. Finzi, F. Pirri, M. Pirrone, M. Romano, and M. Vaccaro. Autonomous mobile manipulators managing perception and failures. In *AGENTS'01, The Fifth International Conference on Autonomous Agents*, 2001.

[10] A. Finzi, F. Pirri, and R. Reiter. Open world planning in the situation calculus. In *Proceedings of AAAI-2000, Seventeenth National Conference on Artificial Intelligence*, 2000.

[11] H. Gaifman. Concerning measures in first-order calculi. *Israel Journal of Mathematics*, 2:1–18, 1964.

[12] K. Z. Haigh and M. M. Veloso. Interleaving planning and robot execution for asynchronous user requests. *Autonomous Robots*, 5(1):79–95, 1998.

[13] J. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46:311–350, 1990.

[14] J. M. Keynes. *A Treatise on Probability*. Macmillan, London, 1921.

[15] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[16] D. McDermott. Probabilistic projection in planning. In *Spatial and Temporal Reasoning. Dordrecht*. Dordrecht: Kluwer Academic, 1997.

[17] J. Pinto, A. Sernadas, C. Sernadas, and P. Mateus. Nondeterminism and uncertainty in the situation calculus. In *Proceedings of the FLAIRS'99 - the 12th International Florida AI Research Symposium*, pages 454–460. AAAI Press, 1999.

[18] F. Pirri and R. Reiter. Some contributions to the metatheory of the situation calculus. *ACM*, 46(3):325–362, 1999.

[19] D. Poole. Decision theory, the situation calculus, and conditional plans. *Linköping Electronic Articles in Computer and Information Science*, 3(8), 1998.

[20] R. Reiter. *Knowledge in Action*. MIT press. To appear, www.utoronto.ca/ cogrobo, 1998.

[21] P. Roeper and H. Leblanc. *Probability Theory and Probability Semantics*. University of Toronto Press, 1999.

Heterogeneity in the Coevolved Behaviors of Mobile Robots: The Emergence of Specialists

Mitchell A. Potter,[1] **Lisa A. Meeden** [2] **and Alan C. Schultz** [1]
[1] Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory, Washington, DC 20375 USA
{mpotter,schultz}@aic.nrl.navy.mil
[2] Computer Science Department, Swarthmore College
Swarthmore, PA 19081 USA
meeden@cs.swarthmore.edu

Abstract

Many mobile robot tasks can be most efficiently solved when a group of robots is utilized. The type of organization, and the level of coordination and communication within a team of robots affects the type of tasks that can be solved. This paper examines the tradeoff of homogeneity versus heterogeneity in the control systems by allowing a team of robots to coevolve their high-level controllers given different levels of difficulty of the task. Our hypothesis is that simply increasing the difficulty of a task is not enough to induce a team of robots to create specialists. The key factor is not difficulty per se, but the number of skill sets necessary to successfully solve the task. As the number of skills needed increases, the more beneficial and necessary heterogeneity becomes. We demonstrate this in the task domain of herding, where one or more robots must herd another robot into a confined space.

1 Introduction

Many mobile robot tasks can be more efficiently solved when a group of robots is utilized. Some tasks cannot be solved at all without multiple robots. The type of organization, and the level of coordination and communication within a team of robots affects the type of tasks that can be solved.

In *swarm* approaches, (usually large) groups of robots execute the same simple strategies with no explicit communication. Complex group behaviors emerge from the simple interactions among the robots. Examples of this include flocking behaviors. In *cooperative* approaches, (usually smaller) groups of robots can have different strategies, allowing some of the robots to become specialists in solving parts of the task—that is, robots can assume roles. Recently, the term *collaborative* has been used to indicate cooperative approaches where the robots explicitly communicate their intent to one another. One important issue in multi-agent robotics is to understand when a particular approach is appropriate for a given task, that is to understand the relative power of each approach.

This paper will examine the tradeoff of homogeneity versus heterogeneity in the control systems by allowing a team of robots to coevolve their high-level controllers given different levels of difficulty of the task. In the homogeneous case, we will restrict the robots to using the same control structure, i.e. only one high-level controller is evolved, which all robots will use. In the heterogeneous case, robots will be allowed to coevolve separate high-level controllers, thus enabling the emergence of specialists. Our hypothesis is that simply increasing the difficulty of a task is not enough to induce a team of robots to create specialists. The key factor is not difficulty per se, but the number of skill sets necessary to successfully solve the task. As the number of skills needed increases, the more beneficial and necessary heterogeneity becomes.

Experiments were conducted within a simulation model of the task domain. The task chosen for these experiments is *herding*, where a group of robots must force another robot into a confined space. The robots are Nomad 200s which are modeled using the TeamBots system [Balch, 1998b]. We show that simply increasing the difficulty of the task alone does not necessarily require heterogeneous control systems, but that introducing a predator into the environment induces the evolution of specialists for defending against the predator.

In the next section we will describe related work. In Section 3, we will describe the task domain and how the complexity of the task can affect coevolved behaviors. The use of a neural network as a high-level controller, and the evolutionary algorithm used for learning will be described in Section 4. We will then describe the experimental methodology and the results in Section 5, followed by our conclusions and a description of future work.

2 Related Work

Evolution of robotic shepherding behaviors was first described in [Schultz *et al.*, 1996], although only one shepherd was involved, and therefore multi-robot coordination was not required. Much has been written about evolution (and co-

evolution) of robot behaviors (see [Mataric and Cliff, 1996; Meeden and Kumar, 1998] for an overview). Several articles have attempted to lay out taxonomies and general issues in multi-robot coordination [Dudek *et al.*, 1993; Cao *et al.*, 1997].

A number of researchers have explored heterogeneity at the hardware level by equipping members of a robot team with different sets of sensors or effectors [Cao *et al.*, 1997; Parker, 1999]. Other researchers have utilized teams of similar agents, and have instead explored heterogeneity at the behavior level [Balch, 1998a; Bongard, 2000; Good, 2000]. Our research takes the latter approach in that each herding agent has the same physical capabilities, but can develop unique control strategies through coevolution.

It remains an open question as to what kinds of tasks warrant a heterogeneous approach. Homogeneous teams have one clear advantage—there is built-in redundancy. If a team member fails for any reason, the rest of the team can still go on and be successful. However, as a task increases in difficulty, division of labor or specialization may become essential for success.

In doing reinforcement learning studies, Balch found that diversity within a team is not always desirable [Balch, 1998a]. Certain kinds of tasks, such as foraging, were solved more easily with a homogeneous approach. Balch speculated that any domain in which an individual agent could reasonably perform the task alone, is well suited for a homogeneous approach. Other domains, such as soccer which seems to require a variety of agent types, were more easily solved with a heterogeneous approach.

One result which is somewhat inconsistent with Balch's conclusions, is Luke's experiences developing a genetic programming based softbot soccer team for RoboCup97 [Luke, 1998]. The main goal of his work was to produce the best team possible in a limited amount of time. Luke developed both homogeneous and heterogeneous teams. However, the heterogeneous teams had a larger search space and thus converged much more slowly. By competition time, the homogeneous teams had the advantage and were entered. Luke predicts that given enough evolution time though, the heterogeneous teams might ultimately win out.

Balch has suggested that tasks that cannot be reasonably solved by a single agent should lead to heterogeneity. Bongard focuses more on the domain rather than the agent and argues that decomposable domains should lend themselves more readily to heterogeneity [Bongard, 2000]. Like Balch, he found that a homogeneous team was more successful at foraging than a heterogeneous one. Yet, it is not clear that this supports his hypothesis. Foraging can be easily decomposed as Balch did in his hand-coded control cases. For example, the domain can be decomposed into separate territories or the targets can be decomposed into particular types.

Our hypothesis is that the need for heterogeneity depends on the number of skill sets required by the domain. The herding domain is a good environment for testing this hypothesis as well as those of Balch and Bongard. We can increase the difficulty of the task along a number of dimensions without requiring new skill sets. We can also increase the number of skill sets required by adding a predator to the environment.

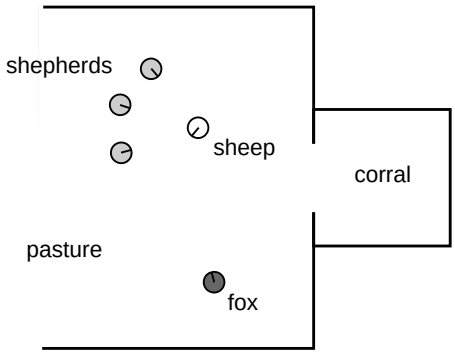


Figure 1: Herding domain

3 Task Domain and Complexity

The herding environment shown in Figure 1 consists of a 37×37 foot pasture that is fenced on three sides, with a smaller enclosed corral on the right. The herding task requires that a group of robots (the *shepherds*) force another robot (the *sheep*) into the corral. The sheep does not want to enter the corral, but instead wants to escape through the unfenced side of the pasture. To further complicate matters, in some experiments there is a predator robot (the *fox*) that attempts to kill the sheep by approaching within a certain distance.

The sheep's behavior is a fixed strategy that causes the sheep to avoid the approach or contact of other robots and obstacles, with additional drives to avoid the corral and to seek escape through the unfenced side of the pasture. The fox's behavior is also fixed, and causes it to attempt to approach the sheep while avoiding the shepherds and other obstacles. If the fox is able to get within a certain distance of the sheep, the sheep dies and the trial is over. The strategies of the shepherds are implemented via high-level neural network controllers which are evolved as described in Section 4.

The complexity of this domain can be controlled along several dimensions. For example, the degree to which the sheep avoids the corral and seeks an escape from the fenced pasture can be increased, the predatory fox can be included, the radius around the sheep in which the fox kills the sheep can be increased, and we can increase the number of sheep. Given the speed and turning rates of the sheep and shepherds, a single shepherd alone can force the sheep into the corral if the sheep only avoids obstacles (including the shepherds). However, if the sheep aggressively avoids the corral and seeks escape then a minimum of two shepherds are required to accomplish this task. By introducing a fox into the environment, a minimum of three shepherds is required.

As a performance task, success is measured by the ability of the shepherds to get the sheep into the corral. The task has failed if the sheep escapes from the pasture, is killed by the fox, or if a time limit is exceeded. The learning task is for the shepherds to evolve neural controllers that allow them to succeed in the performance task. The fitness measure as used by the evolutionary learning method is given in Section 4.

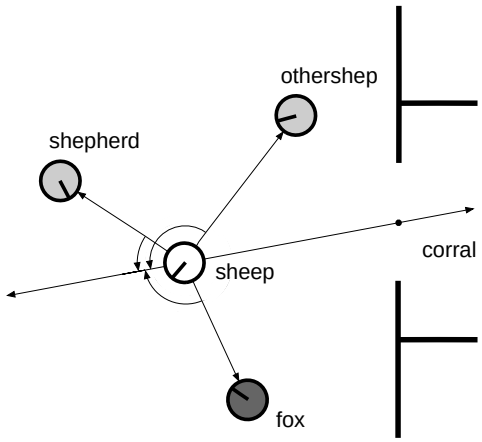


Figure 2: Derivation of sensors used by the neural controller

4 Evolution of Neural Controllers

The high-level controller is implemented with a feed-forward neural network with one hidden layer. The weights of the neural network are evolved using a particular evolutionary algorithm known as an *Evolution Strategy* [Rechenberg, 1964]. As will be seen in a moment, the strategy being evolved represents a high-level behavior, specifically, the neural network generates a goal point at each time step to which the robot will navigate. The lower-level behaviors such as collision avoidance and local navigation are built into the behavior and are not learned.

4.1 Sensors and Actions

Each shepherd is controlled by a neural network that maps its current sensors to a new position to be obtained by the shepherd. This mapping occurs at a 10 Hz rate. Each shepherd has vision sensors that represent the range and bearing to other agents in the environment. In particular, a shepherd can detect the sheep, the closest other shepherd, assuming there is at least one additional shepherd in the environment, and the fox, if present.

These shepherd-based sensor values are translated to ranges and bearing egocentric to the sheep as illustrated in Figure 2. The bearings are relative to the sheep's angle to the corral. Specifically, a polar coordinate system is used in which the pole is centered on the sheep and the polar axis passes through a point at the center of the opening of the corral. All angular measurements are relative to this polar axis. An agent's position is then defined as $(r, \pi - \theta)$, that is, the range is the length of the radius vector from the sheep to the agent, and the bearing is the supplement of the polar angle of the radius vector. The supplement of the polar angle is used so that if an agent is directly behind the sheep with respect to the corral, its bearing will be 0 degrees.

The input to the neural controller includes the following:

1. *shepherd_b:* The bearing from the sheep to the shepherd under control. Present in all experiments.
2. *shepherd_r:* The range from sheep to the shepherd under control. Present in all experiments.
3. *othershep_b:* The bearing from the the sheep to the closest other shepherd. Present in all experiments with two or more shepherds.
4. *othershep_r:* The range from the sheep to the closest other shepherd. Present in all experiments with two or more shepherds.
5. *fox_b:* The bearing from the sheep to the fox. Present in experiments with a predator.
6. *fox_r:* The range from the sheep to the fox. Present in experiments with a predator.

In addition to the vision sensors used by the neural controller, the shepherd also has sonar sensors which are used by the fixed, lower-level behaviors to avoid collisions with other objects, although the distance in which the shepherd will avoid the sheep is less than the distance at which the sheep will avoid the shepherd, allowing the shepherd to be able to herd the sheep.

The output of the neural controller is the range and bearing to the new position to which the robot is to navigate in the same coordinate system as described above. These values are translated to a coordinate system egocentric to the shepherd under control, and input into a motor schema (see [Arkin, 1989]) to produce a linear attraction to the target position. This motor schema is combined with motor schema for obstacle avoidance and stochastic noise into an assemblage which controls the robot via turn and translation rate commands.

4.2 Neural Network Controller

We use a simple two-layer feed-forward neural network topology as shown in Figure 3. Nodes are implemented using a standard sigmoid centered at 0 as follows:

$$f(z_i) = \frac{1}{1 + \exp(-z_i)} - 0.5 \tag{1}$$

$$z_i = \sum_j w_{ij} o_j, \tag{2}$$

where o_j is the output of node j, and w_{ij} is the weight on the connection from node j to node i. All input nodes have weighted connections to all hidden nodes, and all hidden nodes have weighted connections to all output nodes. The network shown is the most complex case, which is used in the experiments with two or more shepherds and a predator. In the experiments with two shepherds and no predator, the number of input nodes is reduced to 5, and in the experiment with one shepherd and no predator, the network is further reduced to 3 input nodes and 3 hidden nodes.

The network accepts real-valued inputs corresponding to the sensors described in the previous section. An additional input is clamped to the value 1.0 in order to provide a learnable bias for each node in the hidden and output layers. The target range output is converted to a value between 0.0 and 10.0 units in simulation, which corresponds to the full 37 foot width of the pasture in the real world, and the target bearing output is converted to a value in the range $(-\pi, \pi)$.

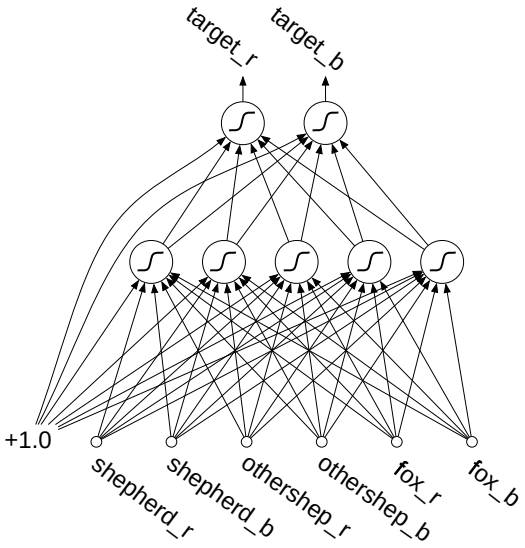


Figure 3: Neural network used as high-level controller

4.3 Evolution of Controllers

The evolutionary algorithm we use to evolve the connection weights for the neural controllers is a $(\mu + \lambda)$ evolution strategy (ES) as described by Bäck and Schwefel [1993], with $\mu = 10$ and $\lambda = 100$. Each individual consists of a real-valued vector of connection weights and a companion vector of standard deviations used by the mutation operator as described below. This class of evolutionary algorithm was introduced in Germany by Rechenberg [1964] for numerical optimization, and variants such as the $(\mu + \lambda)$–ES, originally developed by Schwefel [1981], are a good choice when the problem solution is naturally represented as a vector of real-valued numbers, as is the case when evolving neural network connection weights.

A $(10 + 100)$–ES begins with a population of 10 individuals, and generates 100 children by selecting uniformly from this population and mutating each child. The 100 children and their 10 parents are then evaluated by applying each of them in turn to the target problem, and the 10 individuals with the highest fitness become the next generation of parents. Mutation of an individual $\vec{x}$ with k genes and a companion vector of standard deviations $\vec{\sigma}$ consists of tweaking each gene x_i, for $i = 1, \ldots, k$, according to a normal distribution with mean zero and standard deviation σ_i as follows:

$$x_i' = x_i + \mathrm{N}(0, \sigma_i). \tag{3}$$

Furthermore, the standard deviations $\vec{\sigma}$ are themselves adapted as follows:

$$\sigma_i' = \sigma_i \exp(\tau' \mathrm{N}(0,1), \tau \mathrm{N}_i(0,1)), \tag{4}$$

where

$$\tau = \frac{1}{\sqrt{2\sqrt{k}}} \tag{5}$$

$$\tau' = \frac{1}{\sqrt{2k}}. \tag{6}$$

The elements of $\vec{\sigma}$ are initialized to 1.0 and are restricted to the range $(0.01, 1.0)$. In practice, the standard deviations approach the lower limit of this range over time, which has an effect much like that of simulated annealing.

In the experiments where heterogeneous control systems are evolved, we use the architecture for coevolution developed by Potter and De Jong [2000]. This architecture models an ecosystem consisting of two or more species. As in nature, the species are genetically isolated—meaning that individuals only mate with other members of their species. Mating restrictions are enforced simply by evolving the species in separate populations. The species interact within the herding domain as described below and have a cooperative relationship.

To evaluate an individual from one of the coevolving species given heterogeneous control systems, we construct a neural control system using the individual's genes as connection weights, and assign the resulting control system to one of the shepherds. We then select the current best individual from each of the other species and similarly construct neural control systems from them for assignment to the other shepherds. The shepherds are then set to work on the herding task. Alternatively, we could organize the best shepherds into multiagent squads and assign each squad to a different neural control system. In the experiments where purely homogeneous control systems are evolved, the ecosystem consists of only one species. To evaluate one of these individuals, a neural control system is constructed from that individual's genes and assigned to *all* the shepherds, that is, each shepherd will be controlled by an identical neural network.

In this current study, evaluations are done in simulation. Each evaluation consists of 10 trials in which the shepherds herd the sheep until it is corralled, killed by the fox, escapes from the pasture, or 2.5 simulated minutes have expired. The cumulative distance of the sheep from the corral is measured at the rate of 10 Hz throughout the trial. If the trial ends early due to the sheep escaping or being killed, we continue to accumulate the full pasture-width distance until the clock expires. The fitness of an individual is taken to be the final cumulative distance averaged over the 10 trials. The worst possible fitness is 14,768, which would result if the sheep immediately initiated a turn towards the left side of the pasture and escaped without any interference from the shepherds. The best possible fitness is 976, which would result if the sheep set a course directly towards the corral at maximum speed. The ES will seek to minimize this measure.

5 Results

In order to test our hypothesis, we vary the complexity of the task in several dimensions using both homogeneous and heterogeneous multi-agent approaches.

We begin with the simplest case of one shepherd herding one passive sheep that just avoids obstacles. This is essentially a reimplementation (in simulation) of earlier work done by Schultz *et al.* [1996]. We repeat this earlier experiment to validate the design of our neural controller and evolutionary learning method. The results are shown in Figure 4 along with results from a second experiment in which the sheep

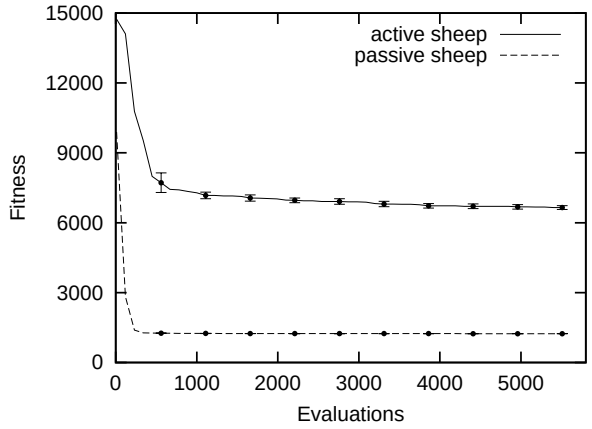


Figure 4: One shepherd herding one sheep

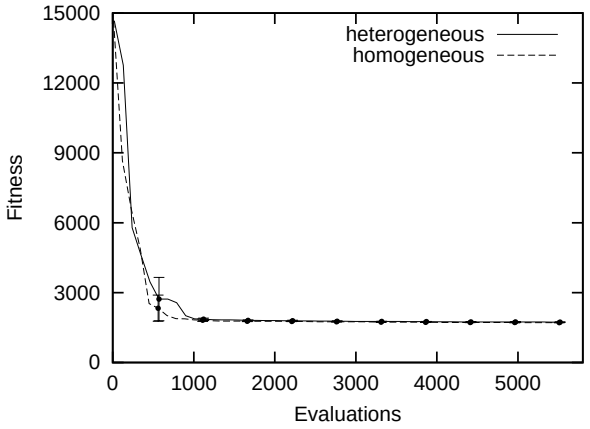


Figure 5: Two shepherds herding one sheep

actively seeks escape from the pasture and avoids the corral. The curves in the graph represent the mean fitness of the best individual seen so far averaged over 10 separate runs. Overlaid on the curves at increments of 5 generations (550 evaluations) are 95-percent confidence intervals on the mean. Although these control systems were evolved for 50 generations (5,500 evaluations), a controller with near optimal behavior on this simple task was easily evolved in as little as 2 generations. We also observed simulated robots solving this task using the best neural control system from the final generation of evolution, and it is clear that they are exhibiting behavior very much like the behavior evolved in the earlier work by Schultz *et al.*. Specifically, the robot positions itself directly behind the sheep with respect to the corral, and herds the sheep by moving towards it and triggering its obstacle avoidance behavior. The shepherd makes subtle swings from left to right to counter irregularities in the movement of the sheep. In contrast, the more complex case involving the sheep seeking freedom (hereby referred to as the *active* sheep) does not appear solvable with a single shepherd, as indicated by the learning curve flattening out at a very poor level of fitness.

The result of adding a second shepherd to the more difficult task of herding an active is shown in Figure 5. We compare both the evolution of two shepherds using homogeneous control systems, and two shepherds using heterogeneous control systems. Although it took slightly fewer evaluations to evolve good homogeneous control systems, adequate behavior was also evolved in the case of heterogeneous control. Due to the significantly smaller search space of homogeneous controllers, it is not surprising that when good homogeneous solutions exist it is easier to find them. When observing robots solving this task using the best neural control systems from the final generation of evolution, we see that in both cases (heterogeneous and homogeneous) the pair of robots assume positions behind the sheep, but slightly to the left and right. This counters the strong tendency of the sheep to slip by the shepherds and escape from the pasture. This is clearly a cooperative approach on the part of the shepherds, and it shows

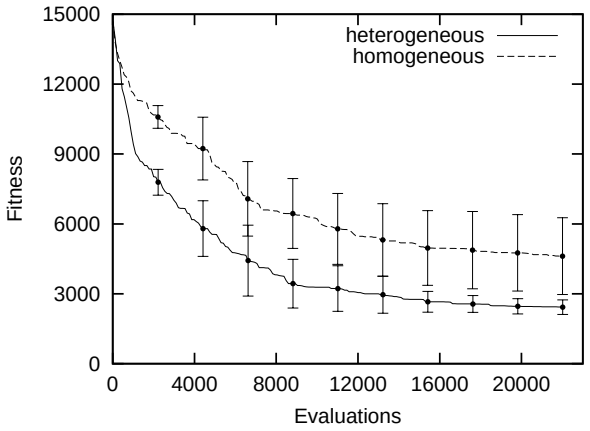


Figure 6: Three shepherds herding one sheep with fox

the need for cooperation is not alone sufficient to warrant the use of heterogeneous control.

Finally, we add a predator to the environment, in the form of a fox that seeks to kill the sheep. Now the shepherds require two skills—the ability to herd the sheep and the ability to keep the fox a safe distance away from the sheep. To encourage specialization, we initially position two shepherds behind the sheep, and a third shepherd is positioned between the sheep and the fox. As before, we compare the evolution of shepherds using homogeneous control systems with the evolution of shepherds using heterogeneous control systems. In the case of heterogeneous control, the two shepherds positioned behind the sheep are controlled by one neural network and the shepherd positioned closer to the fox is controlled by a second neural network. The results from this experiment are shown in Figure 6. This task is much more difficult than the previous ones, so we evolved these control systems for 200 generations (22,000 evaluations). The graph clearly shows the superiority of heterogeneous control on this task.

Control	*Shepherds*	*Sheep*	*Fox*	*Fitness*			*Success Ratio*
				Mean	*Minimum*	*Maximum*	
homogeneous	1	passive	no	1273.9 ± 2.7	1205.3	1356.0	1.000
homogeneous	1	active	no	7592.5 ± 127.3	5865.9	14548.7	0.000
homogeneous	2	active	no	1808.9 ± 60.4	1636.8	14621.8	0.996
heterogeneous	2	active	no	1865.5 ± 81.4	1645.7	14573.0	0.992
homogeneous	3	active	yes	4336.0 ± 372.8	1669.9	13966.6	0.758
heterogeneous	3	active	yes	3149.2 ± 305.5	1579.8	14153.8	0.874

Table 1: Comparison of the mean fitness and success rate of the best individual from each experiment, evaluated over an additional 500 trials

When we observe the heterogeneous control systems performing this task, we see that the shepherd that is initially positioned between the sheep and fox does indeed exhibit specialized blocking behavior, while learning to herd as well. It begins by moving towards the sheep, keeping its body between the sheep and fox. When it nears the sheep, it sometimes turns to face the fox and performs a quick deflection maneuver before it joins the other two shepherds begins herding. However, most of the time its blocking behavior is more subtle. Close observation reveals that the blocking shepherd maintains slightly more distance between itself and the sheep than the other two shepherds, which enables it to keep the fox at a safe distance from the sheep while still providing some help in herding. Without having to concern themselves as much with the fox, the other two shepherds are able to herd the sheep more directly towards the corral. The homogeneous control systems rely much more on herding the sheep away from the fox, which is not as effective as blocking because it gives the sheep more opportunities to escape from the pasture.

It should be noted that there is a difference in complexity between the heterogeneous and homogeneous control systems. Since the heterogeneous control system utilizes two separate neural networks—one for the pair of shepherds initially positioned behind the sheep, and one for the shepherd positioned between the sheep and fox—a total of 94 connection weights are being evolved, while only 47 connection weights are evolved for the single-network homogeneous control system. To verify that the observed difference in performance is not simply due to this difference in complexity, we evolved a homogeneous control system with a complexity on the order of the heterogeneous system by increasing the number of hidden units to 10, which produced a neural network with 92 connection weights. As expected, this homogeneous control system performed much more poorly than the homogeneous control system with only 5 hidden units. Specifically, averaged over 10 runs to 200 generations, the final mean fitness of the more complex homogeneous control system was 7,918, compared with an average fitness of 4,614 for the simpler homogeneous control system.

The results from the previous three graphs are summarized and further supported by Table 1. Here we take the single best individual from the 10 runs of each experiment and apply it to the herding task for an additional 500 trials. We report the mean fitness, along with 95-percent confidence intervals on this mean, the maximum and minimum fitness achieved, and the success ratio, that is, the percentage of trials in which the sheep was actually corralled. The table reinforces our earlier observation that a single shepherd is not capable of herding an active sheep into the corral. However, two cooperating shepherds are sufficient to accomplish this mission. Heterogeneous control systems are not an advantage here, in fact, they perform slightly worse, although a *t*-test on the means of the two-shepherd homogeneous and heterogeneous trials produced a *p*-value of 0.2729, indicating a lack of statistical significance in their difference. Only when the task requires multiple skills (e.g., herding the sheep and blocking the predator) does heterogeneous control perform better than homogeneous control, as indicated by the trials with three shepherds. A *t*-test on the means of the three-shepherd trials produced a *p*-value of 0.0000, clearly showing a statistically significant advantage to using heterogeneous control.

6 Conclusion

In this paper, we have tried to demonstrate that simply increasing the difficulty of a task is not enough to induce a team of robots to create specialists. The key factor is not difficulty per se, but the number of skill sets necessary to successfully solve the task. As the number of skills needed increases—in this study by adding the response to a predator—the more beneficial and necessary heterogeneity becomes.

Although heterogeneous control systems can promote better solutions for many tasks, there is a trade off. Learning (coevolving) a team of homogeneous agents can take much less time, since each evaluation of an individual in the population goes towards all individuals' progress and the search space is smaller. In a heterogeneous group, the available CPU time during evolution must be divided among the different skill sets.

Ongoing experiments are attempting to more generally determine the properties that dictate the type of approach that is appropriate. In addition, results will be duplicated on the physical Nomad 200 robots to show that the simulation results hold on the actual robots.

Acknowledgments

This work was supported by the Office of Naval Research under work request N0001401WX20073.

References

[Arkin, 1989] Ronald C. Arkin. Motor schema-based mobile robot navigation. *The International Journal of Robotics Research*, 8(4):92–112, 1989.

[Bäck and Schwefel, 1993] Thomas Bäck and Hans-Paul Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[Balch, 1998a] Tucker Balch. *Behavioral Diversity in Learning Robot Teams*. PhD thesis, Georgia Institute of Technology, 1998.

[Balch, 1998b] Tucker Balch. Integrating robotics research with javabots. In *Working Notes of the AAAI-98 Spring Symposium, Stanford, CA*, 1998.

[Bongard, 2000] Josh C. Bongard. The legion system: A novel approach to evolving heterogeneity for collective problem solving. In *Genetic Programming: Third European Conference*, pages 25–37. Springer-Verlag, 2000.

[Cao *et al.*, 1997] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:7–27, 1997.

[Dudek *et al.*, 1993] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for swarm robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 441–447, 1993.

[Good, 2000] Benjamin McGee Good. Evolving multi-agent systems: Comparing existing approaches and suggesting new directions. Master's thesis, University of Sussex, 2000.

[Luke, 1998] Sean Luke. Genetic programming produced competitive soccer softbot teams for RoboCup97. In John Koza, editor, *Proceedings of the Third Annual Genetic Programming Conference*, pages 214–222. Morgan Kaufmann, 1998.

[Mataric and Cliff, 1996] M. Mataric and D. Cliff. Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems*, 19:67–83, 1996.

[Meeden and Kumar, 1998] Lisa A. Meeden and Deepak Kumar. Trends in evolutionary robotics. In L.C. Jain and T. Fukuda, editors, *Soft Computing for Intelligent Robotic Systems*, pages 215–233. Physica-Verlag, New York, NY, 1998.

[Parker, 1999] L. E. Parker. Adaptive heterogeneous multi-robot teams. *Neurocomputing*, 28:75–92, 1999.

[Potter and De Jong, 2000] Mitchell A. Potter and Kenneth A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[Rechenberg, 1964] Ingo Rechenberg. Cybernetic solution path of an experimental problem. Library Translation 1122, August 1965. Farnborough Hants: Royal Aircraft Establishment. English translation of lecture given at the Annual Conference of the WGLR, Berlin, 1964.

[Schultz *et al.*, 1996] A. C. Schultz, J. J. Grefenstette, and W. Adams. Robo-shepherd: Learning complex robotic behaviors. In M. Jamshidi, F. Pin, and P. Dauchez, editors, *Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*, pages 763–768. ASME Press, 1996.

[Schwefel, 1981] H.-P. Schwefel. *Numerical Optimizaion of Computer Models*. Chichester: Wiley, 1981.

Agent-Based Control for Object Manipulation with Modular Self-reconfigurable Robots

Jeremy Kubica and **Arancha Casal** and **Tad Hogg**
Xerox Palo Alto Research Center
Palo Alto, CA 94304

Abstract

We demonstrate multiagent control of modular self-reconfigurable (MSR) robots for object manipulation tasks and show how it provides a useful programming abstraction. Such robots consist of many modules that can move relative to each other and change their connectivity, thereby changing the robot's overall shape to suit different tasks. We illustrate this approach through simulation experiments of the TeleCube MSR robot system.

1 Introduction

Modular self-reconfigurable (MSR) robots [37, 31, 38, 32, 25, 29, 30, 9] consist of many identical modules that can attach and detach from one another to change their overall connectivity. Each module has processing, sensing and actuation capabilities. The main characteristic of MSR robots is their ability to change shape to suit their task. While such shapes are unlikely to be as effective at particular tasks as special purpose tools (e.g., the modules are unlikely to have the same strength as, say, a hammer), they provide adaptability for a wide range of tasks in unpredictable environments. For instance, changing shape can provide a variety of gaits for locomotion, e.g., wheels for flat surfaces, snakes for small tunnels, etc. Robot shapes matching object contours can also help provide precise manipulation.

Beyond the challenge of building many such modules, their tight physical interactions and many degrees of freedom pose a difficult control problem. Moreover, modules constructed of micromachines (MEMS) [7] or even smaller devices [14, 11, 10] are likely to have limited capabilities and be prone to some failures or incorrect assembly. Thus the robust functioning of the robot requires a control architecture that tolerates such limitations, as has also been proposed for computing with molecular-scale devices [18].

One control approach would be to precisely specify the desired locations of all the modules, and perform a combinatorial search to find appropriate module motions according to some criterion, such as minimizing moves or power consumption. With many modules, such searches are generally intractable, but can be solved approximately using heuristics [31, 38, 29, 25, 32, 9, 20]. Even so, a high-level controller often lacks enough information on the modules' statusor the task environment to define a precise shape, e.g., when grasping an object whose detailed size or shape is unknown.

Instead, a structure with the general properties required for the task is often sufficient. When the properties are determined mainly by the local environment for each module, agent-based local control often finds a suitable shape without any need to precisely specify each module's position [4]. A multiagent approach to general behaviors, such as locomotion and object manipulation, simplifies the design of higher-level control by presenting a useful programming abstraction to a centralized planner or within levels of a sequence of increasingly complex behaviors [6]. That is, higher level control programs can determine the choice of local rules for the agents without detailed concern of the particular motion primitives of the modules or situation of each agent individually. For example, by switching among a set of available behaviors or modifying parameters used by the agents' rules.

Agent-based architectures readily match control to physical phenomena dominant at the different scales relevant to MSR robots with many modules. For example, micromachined robots [12] are dominated by friction and other surface forces rather than gravity. Even smaller structures [21] are subject to randomly fluctuating forces, i.e., Brownian motion. Biological structures provide numerous other examples [36]. In such cases, different agent types could be responsible for individual modules, small groups of modules and so on, forming a hierarchical or multihierarchical correspondence between the robot's physical structures and its environment, task and the controlling software [19].

In related work, multiagent control helps teams of robots cooperating to achieve a common task [35, 8, 33, 17, 24]. These teams usually consist of independently mobile robots that are not physically connected and have little or infrequent physical contact. In most current modular robot systems the modules remain attached to one another forming a single connected whole, and giving rise to a number of tight physical motion constraints that do not apply to teams of separate robots. On the other hand, the physical contact between modules allows them to locate their neighbors without complex sensory processing as would be required, for example, to visually identify a physically disconnected member of a team. Hence the techniques for coordinating teams address somewhat different requirements than those of MSR robots.

The key issue of relating local rules to global behavior also

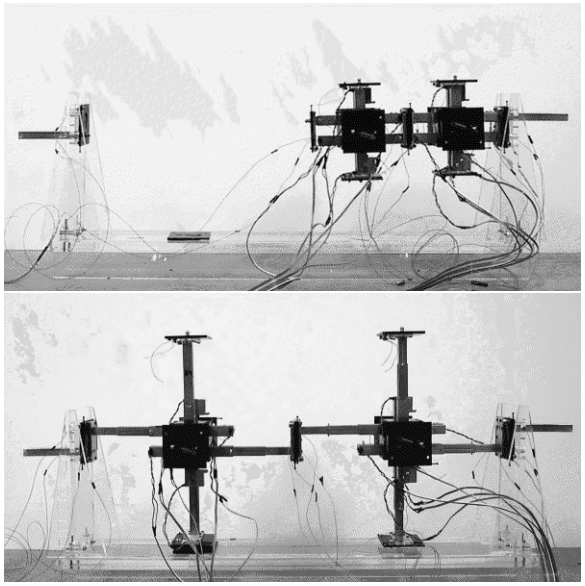


Figure 1: The TeleCube hardware: two modules shown with arms retracted and expanded. The current modules are about 15cm in diameter when fully contracted.

arises in artificial life [13] which concentrates on how complex natural organisms achieve sophisticated crowd behaviors or deal with abstract agents that currently can not be physically constructed [15, 16, 1]. Alternatively, biological and chemical techniques [5, 27] give complex shapes from local behaviors, but can not yet produce programmable robots.

Thus while multiagent systems have been applied to several related tasks, identifying appropriate agent behaviors for MSR robots remains an open problem. In the remainder of this paper, we describe the TeleCube modular robot and simple agent behaviors that provide basic object manipulation capabilities. These agent behaviors, in turn, abstract away the detailed nature of the module motion capabilities, providing a simpler programming model for higher level control.

2 The TeleCube Modular Robot

This section describes the modular robot we used, its simulator and the agent-based control technique.

2.1 Hardware

TeleCube is a new MSR robot where each module is a cube that can prismatically extend each of its six faces independently up to a factor of two times its fully retracted configuration (Fig. 1). A 2D analogue was previously developed at Dartmouth [32]. Each module can communicate with its immediate neighbors in all six directions. When a module extends an arm in contact with an object, it can exert a force on the object. This design allows modules bound inside a group to move, so TeleCube can perform tasks inside the aggregate, such as internal object manipulation and density changes.

2.2 Simulator

To examine behavior with more modules than currently feasible to construct, we simulated the TeleCube system. The simulator, written in Java, accounted for the motions of the modules in three dimensions and typically ran with hundreds of modules. The simulator performs a series of steps. For each step, a random ordering of the modules is selected, thereby simulating asynchronous module actions. Using this ordering, each module is given a "turn" in which to decide its behavior and perform any motion it selects.

For vertical motion, the simulator included the effects of gravity, as appropriate for relatively large scale objects such as the current TeleCube modules. For smaller scale modules, viscous and frictional forces will become more important and could be included instead of gravity.

The simulator makes several simplifying assumptions. First, the module arms have only two states: fully expanded or contracted. Second, the arms are infinitely rigid. Both of these assumptions simplify alignment of modules after a move. In practice, such alignment can be done with low-level feedback control or engineering the connectors on the module arms to guide slightly misaligned parts into alignment. Lastly, the simulator assumed enough friction between the module and the floor to eliminate any sliding. We use a motion primitive to ensure correct module movement by including required low-level actions such as disconnection from modules perpendicular to the motion direction, connection with any new neighbors and possible arm contraction while a neighbor extends its arm [26]. A global connectivity check insures disconnecting from a neighbor does not separate the ensemble. This check may be easily preformed in hardware, and was simulated with a depth-first search of the modules.

2.3 Agent-Based Control

Our approach views each module as an independent agent executing a simple program, consisting of a finite-state machine (FSM). This approach uses the distributed, homogenous and networked nature of MSR robots and scales well as the number of modules increases. There is no central control processor or designated leader module. Global motion results, or "emerges", from purely local control rules executed at every module and through communication between neighboring modules alone. These behaviors can then be turned on and off by a higher-level system to give overall control for a sequence of tasks.

The FSM control consists of simple if-then rules which may act probabilistically. Modules switch among "states" that determine their behaviors and communicate by sending and receiving "messages". Randomization helps prevent modules from becoming indefinitely stuck in unproductive configurations. It also prevents all modules from moving at once, potentially disconnecting the structure or leading to oscillations due to synchronous activity in which modules respond based on a system state that no longer holds if too many neighbors move simultaneously [22].

Messages are the means of global communication among modules. Messages are propagated through the system in a distributed breadth-first fashion and can be modified as they move from one module to another. A "scent" is a message

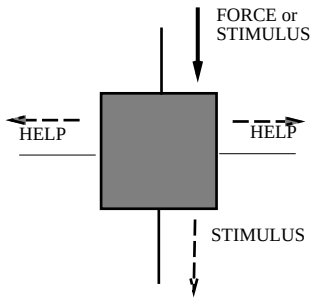


Figure 2: Message passing from a **column** module. Help messages are also sent in the other two directions: into and out of the page.

consisting of a numerical value that is slightly adjusted up or down as it passes through the modules [4]. Propagating scents form global gradients that can guide module motions. For instance, a scent indicating the distance to some location (e.g., a module under high stress) could consist of a single integer value a module sends to its neighbors. A module increments the minimum value of all such messages it receives and sends the new value to its neighbors. Using the *minimum* function to combine values from neighbors allows the scent to provide a useful gradient in spite of many cycles in the graph of links among the modules. These messages can have a maximum number of iterations to prevent them passing among modules indefinitely. Once these messages have passed through a region of modules, the difference in recorded values between one module and a neighbor is a local scent gradient.

Biological concepts, such as scents and hormones, have been applied to MSR robots for reconfiguration, grouping, synchronization and locomotion [4, 26, 34]. Extending these concepts to consider gradients with TeleCube's ability to move modules within a structure allows object manipulation as described below.

We assume the modules are small compared to the size of objects in their environment, allowing many modules to act on them at once. Furthermore, we take module motions to be fast compared to the other object speeds, so numerous module updates take place with little change in the objects' positions. Since modules move slower than they communicate, the response to messages does not introduce oscillating behavior [23] or irregular growth, and the intermediate configurations are well-balanced at all times.

3 Results

This section presents the rules and resulting behaviors for two tasks involved in object manipulation.

3.1 Growing toward an object

Our first task is a dynamic supporting structure in which modules under pressure signal others to add their support, reducing the weight pressure (i.e., force per supporting module) and providing more uniform support for the object. This application requires a pressure sensor in each module. Control is handled through four different states: **relaxed**, **column**, **expanding**, and **reset** and three different messages: *stimulus*, *help*, and *reset*.

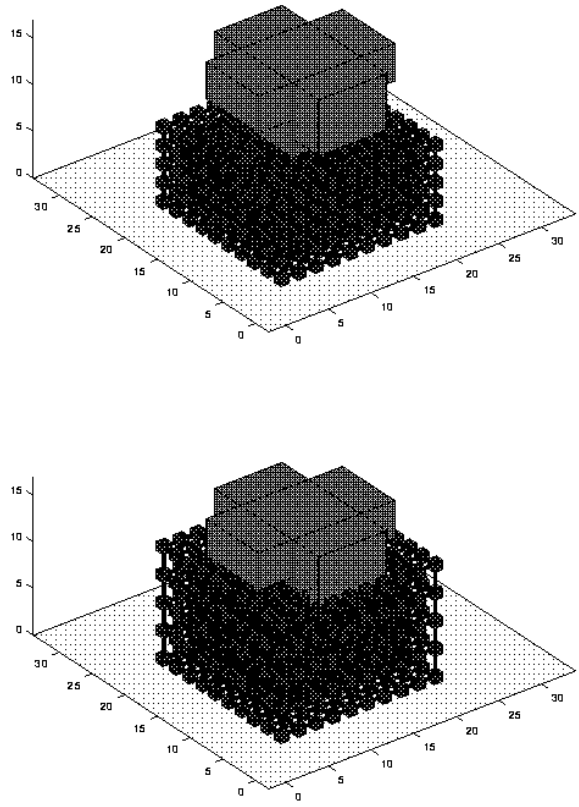


Figure 3: Growing up toward an arbitrary-shaped object. This example required 30 simulation steps.

All modules begin in a **relaxed** state. When a module receives an external force stimulus, it sends a *stimulus* message to the modules along the direction of the force (one of the module's six faces). Upon receiving the message the module transitions to the **column** state and effectively forms a rigid support column. This column sends *help* messages to neighboring modules, which ask them to expand toward the object and provide support. The remainder of this section provides details of the behaviors.

Behavior for one turn in the **relaxed** state is:

- If a *stimulus* message is received, transition to **column** state and end turn.
- If a *help* message is received, record the direction from which the message came (the "help direction"), transition to **expanding** state, and end turn.
- If contact with an object is detected, transition to **column** state and end turn.
- Otherwise, when no message is received, adjust module density in all six directions as follows: For each direction, if the module is farther from or closer to a neighbor than one fully extended arm length, move towards or

away from the neighbor, respectively, with a probability of 50%. This motion tends to restore a uniform density.

Modules in the **expanding** state move in the direction where additional support is required (the "help direction"). However, to insure that the bottom level remains connected, the last module in each column (as determined by the direction of the original force stimulus) does not move or disconnect from its neighbors. Behavior of the **expanding** state is:

- If a *help* message is received, update help direction.
- If a *stimulus* message is received, transition to **column** state and end turn.
- If the force stimulus is removed, transition to **reset** state and end turn.
- If contact with an object is detected, transition to **column** state and end turn.
- Otherwise, send *help* messages in all six directions and, with 50% probability, move in the direction of required help, unless the opposite direction has no neighbor.

Modules in the **column** state remain rigid and stationary, providing support for an object. They emit *help* and *stimulus* scents to the appropriate neighbors, effectively calling for help and telling the others in the column to hold fast. Behavior of the **column** state is:

- If the force stimulus is removed, transition to **reset** state and end turn.
- Send *stimulus* messages to the modules above and below as determined by the direction of stimulus.
- Send *help* messages to modules perpendicular to the direction of stimulus, as shown in Fig. 2.

The **reset** state is used when the object is removed so modules return to their original configuration. A module in the **reset** state sends the *reset* message to all neighbors in the direction along which the force previously acted and transitions to the **relaxed** state.

3.2 Manipulating an object

The TeleCube architecture allows manipulating objects inside a group of modules. This can be accomplished by growing around a supported object or, more generally, opening gaps within the structure and pushing an object into them.

Internal object manipulation requires translational and rotational motions. In both cases, a module that is not in direct contact with an object moves based on the gradient of messages it receives from its neighbors, determined by comparing the relative numerical values of the "scent" messages. Namely, if possible, a module will move in the direction of the positive gradient. If the module does not sense a gradient, it will try to move so as to restore an "optimal" density of modules in its neighborhood. This means moving away from a neighbor that is too close or toward a distant neighbor.

Internal manipulation uses **shell** and **tissue** states, and one *stimulus* message propagated among neighbors to form a gradient scent. The process starts with a broadcast of the desired

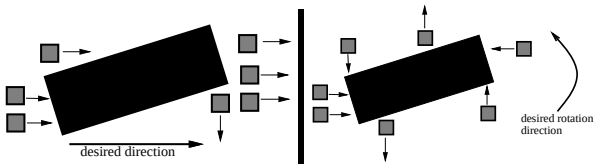


Figure 4: Module motions to move (left) and rotate (right) an object in 2D. Only a few of the modules surrounding the object are shown.

movement direction. Modules on the outside of the structure enter the **shell** state, and do not move under the influence of gradients or density adjustments. Thus, these modules form a rigid support structure off which inside modules can push, i.e., MSR robots can not only form tools shaped to their task, but can also provide dynamic support structures for using those tools. Interior modules enter the **tissue** state.

Translational 2D motion uses a simple stimulus-based response. Given the desired direction of motion, modules in contact with the object use the following rules:

- If a module is in front of the object (i.e., touching it with an arm extending from the module in a direction going against the desired direction of object movement), it releases a negative *stimulus* message and then tries to move away from the object. If possible, it moves in the direction opposite to the arm touching the object. If that direction is blocked by other modules, it tries to move instead perpendicular to that direction. Either motion tends to move the module out of the object's way. Furthermore, the negative *stimulus* message creates a scent gradient among neighbors causing them to also move away and make room for the module to move.
- If a module is behind the object (i.e., touching it with an arm in a direction aligned with the desired direction of object movement), it releases a positive *stimulus* and tries to move toward and push the object. If the module has more than one arm in contact with the object, this rule applies to the arm most closely aligned with the desired direction of motion.

These behaviors are illustrated in Fig. 4(left) and result in modules pushing the object in the desired direction.

Rotation requires the approximate location of the object's center to allow relating individual module forces to the overall torque imposed on the object. Estimating this location via a distributed local algorithm can be difficult if the modules are much smaller than the object and, on the scale of the modules, the object is not smooth. Instead, we assume the higher level controller broadcasts the approximate location of the object's center. Another possibility is to use open-loop force fields to position the object in a known starting configuration, as demonstrated on a micromachined surface [3]. In either case, for the behavior described here we suppose knowledge of the approximate center. That is, any error in the center's specified location is small compared to the object's size, but not necessarily small compared to the size of the modules. The modules act in the same way as the translation modules, except they move away from or towards the object if they are in front of a corner that should be turned

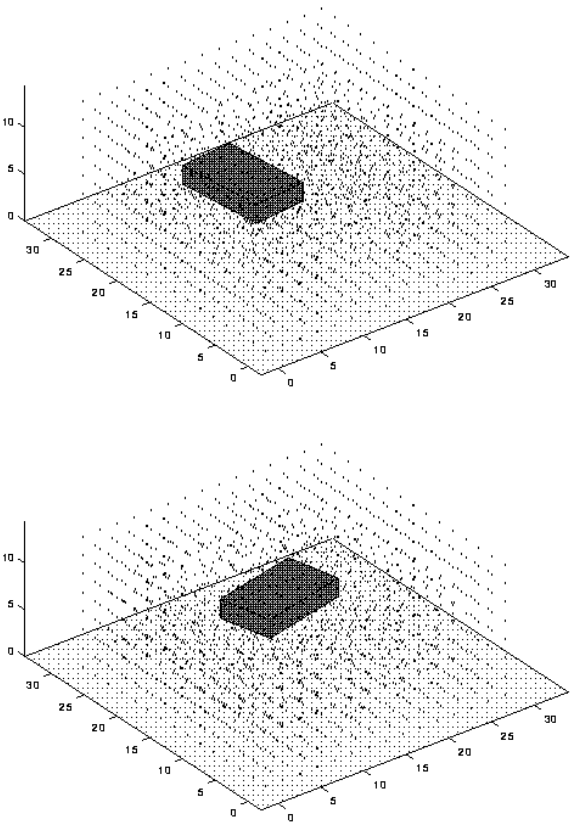


Figure 5: Internally manipulating an object. The modules form a $15 \times 15 \times 8$ structure, and are shown simply as points to allow seeing the internally manipulated object. The full sequence, consisting of 138 simulation steps, shows the object gradually shifted from its initial to final positions in response to instructions to move the object in an "L" shaped path. The figure shows steps 15 and 80.

towards or away from them, respectively, as determined by their relative location to the specified center and the desired direction of rotation. Again, they release positive and negative *stimulus* messages when moving towards and away from the object, respectively, and push the object when they move towards it. This behvaior, shown in Fig. 4(right), gives 2D rotation in a plane.

Fig. 5 includes both types of motion. First the modules are told to move the object in one direction, then to rotate it, and finally move in the perpendicular direction. These instructions cause modules to switch between translation and rotation rules, giving desired motions without the higher-level control specifying individual module motions. We found the local rules could produce a variety of trajectories. The combination of translation and rotation rules allows achieving arbitrary planar position and orientation of the object.

Modules move the object by exerting forces on it. They could also exert compressive forces on the object by simply replacing the negative *stimulus* messages with positive ones so modules on all sides move toward the object. This behavior could be used to push parts together. Shear forces could arise from tangential motion, provided friction between the arms of the modules and the object was sufficient. The modules then can be viewed as a material surrounding the object with forces under programmed control, changing from rigid support to fluid-like flow as needed. This behavior is a useful programming astraction for higher levels of control, similar to programmable force fields in two dimensions [3]. By giving different instructions to different regions, a higher-level control could use the modules to move multiple objects along different paths or, through feedback based on the position of the objects (rather than details of the much larger number of modules), bring objects together.

4 Discussion

This paper described local rules for two object-handling behaviors for TeleCube robots, dynamic support and internal manipulation. Local rules can also reconfigure them for locomotion and navigation [26].

Our results extend prior distributed control [4] of Proteo [38], another MSR robot whose motions differ in two important ways from TeleCube. First, Proteo modules move independently. By contrast, a TeleCube module usually requires cooperative actions from neighbors to push or pull it, as shown in Fig. 1. Second, Proteo modules are rigid and can only move on the external surface created by other modules. Thus, Proteo can not internally manipulate objects unless the control creates holes inside the structure. TeleCube modules can deform and move inside a structure of other modules. The different hardware capabilities of Proteo and TeleCube affect the design and complexity of the local control rules that are appropriate to consider. Thus hardware designs should balance the control complexity and the manufacturing difficulty. Ideally, the hardware should support simple high-level programming abstractions while the control simplifies the required hardware capabilities. Schemes based on local (module-level) control help with this since they are simple and can be easily modified to apply to a wide range of module designs, including Proteo and prismatic robots, such as TeleCube and the Crystalline robot at Dartmouth [32].

Our approach could be combined with other methods, as part of an overall hierarchical control scheme. Moreover, testing a population of agents against variations in the desired task can evolve better behaviors [2]. For instance, with the primitives presented in this paper, genetic techniques could identify methods to modulate the rule parameters and switch among rules to achieve higher-level goals.

As robots with many modules are built, one important question is how module failures affect system behavior. Scaling is another important issue, i.e., increasing the number of modules or decreasing their size, or both. More modules allow greater variety of interactions and statistically more robust randomized behaviors. Smaller modules allow finer scale object manipulation. However, smaller modules take smaller steps resulting in slower manipulation unless they complete each step more rapidly. This observation highlights the importance of balancing the scaling of speed, strength and other module properties [11, 36] as, for example, with protein motors that carry weights much larger than their own [21, 28].

Multiagent control can balance competing goals to help such robots achieve robust behaviors.

References

[1] Harold Abelson et al. Amorphous computing. Technical Report 1665, MIT Artificial Intelligence Lab, August 1999.

[2] Forrest H. Bennett III and Eleanor G. Rieffel. Design of decentralized controllers for self-reconfigurable modular robots using genetic programming. In *Proc. of the 2nd NASA/DOD Workshop on Evolvable Hardware*, July 2000.

[3] Karl F. Bohringer et al. Computational methods for design and control of MEMS micromanipulator arrays. *Computational Science and Engineering*, 4(1):17–29, January-March 1997.

[4] Hristo Bojinov, Arancha Casal, and Tad Hogg. Multiagent control of modular self-reconfigurable robots. In *Proc. of Intl. Conf. on Multiagent Systems (ICMAS2000)*, 2000. An extended version is Los Alamos preprint cs.RO/0006030.

[5] Ned Bowden, Andreas Terfort, Jeff Carbeck, and George M. Whitesides. Self-assembly of mesoscale objects into ordered two-dimensional arrays. *Science*, 276:233–235, 1997.

[6] Rodney A. Brooks. New approaches to robotics. *Science*, 253:1227–1232, September 13 1991.

[7] Janusz Bryzek, Kurt Petersen, and Wendell McCulley. Micromachines on the march. *IEEE Spectrum*, pages 20–31, May 1994.

[8] P. Caloud et al. Indoor automation with many mobile robots. In *Proc. of the Intl. Workshop on Intelligent Robots and Systems*. IEEE, 1990.

[9] Arancha Casal and Mark Yim. Self-reconfiguration planning for a class of modular robots. In *SPIE Symposium on Intelligent Systems and Advanced Manufacturing: Sensor Fusion and Decentralized Control in Robotic Systems*, pages 246–257, 1999.

[10] H. G. Craighead. Nanoelectromechanical systems. *Science*, 290:1532–1535, 2000.

[11] K. Eric Drexler. *Nanosystems: Molecular Machinery, Manufacturing, and Computation*. John Wiley, NY, 1992.

[12] Thorbjörn Ebefors et al. A walking silicon micro-robot. In *Proc. of the 10th Intl. Conf. on Solid-State Sensors and Actuators (TRANSDUCERS99)*, pages 1201–1205, 1999.

[13] Joshua M. Epstein and Robert Axtell. *Growing Artificial Societies*. MIT Press, Cambridge, MA, 1996.

[14] Robert A. Freitas Jr. *Nanomedicine*, volume 1. Landes Bioscience, 1999.

[15] S. Hackwood and G. Beni. Self-organization of sensors for swarm intelligence. In *Proc. of the Conference on Robotics and Automation (ICRA92)*. IEEE, 1992.

[16] J. Storrs Hall. Utility fog: The stuff that dreams are made of. In B. C. Crandall, editor, *Nanotechnology*, pages 161–184. MIT Press, Cambridge, MA, 1996.

[17] Brosl Hasslacher and Mark W. Tilden. Living machines. In L. Steels, editor, *Robotics and Autonomous Systems: The Biology and Technology of Intelligent Autonomous Agents*. Elsivier, 1995.

[18] James R. Heath, Philip J. Kuekes, Gregory S. Snider, and R. Stanley Williams. A defect-tolerant computer architecture: Opportunities for nanotechnology. *Science*, 280:1716–1721, 1998.

[19] Tad Hogg and Bernardo A. Huberman. Controlling smart matter. *Smart Materials and Structures*, 7:R1–R14, 1998. Los Alamos preprint cond-mat/9611024.

[20] K. Hosokawa et al. Self-organizing collective robots with morphogenesis in a vertical plane. In *Proc. of the Conference on Robotics and Automation (ICRA98)*. IEEE, 1998.

[21] Joe Howard. Molecular motors: Structural adaptations to cellular functions. *Nature*, 389:561–567, 1997.

[22] Bernardo A. Huberman and Natalie S. Glance. Evolutionary games and computer simulations. *Proceedings of the National Academy of Science USA*, 90:7716–7718, August 1993.

[23] J. O. Kephart, T. Hogg, and B. A. Huberman. Dynamics of computational ecosystems. *Physical Review A*, 40:404–421, 1989.

[24] Hiroaki Kitano, editor. *Robocup-97: Robot Soccer World Cup I*, volume 1395 of *Lecture Notes in Computer Science*. Springer, Berlin, 1998.

[25] Keith Kotay, Daniela Rus, Marsette Vona, and Craig McGray. The self-reconfiguring robotic molecule: Design and control algorithms. *Algorithmic Foundations of Robotics*, 1998.

[26] Jeremy Kubica, Arancha Casal, and Tad Hogg. Complex behaviors from local rules in modular self-reconfigurable robots. In *Proc. of ICRA2001*, 2001.

[27] Ross J. Metzger and Mark A. Krasnow. Genetic control of branching morphogenesis. *Science*, 284:1635–1639, 1999.

[28] Carlo Montemagno and George Bachand. Constructing nanomechanical devices powered by biomolecular motors. *Nanotechnology*, 10:225–231, 1999.

[29] Satoshi Murata et al. A 3-D self-reconfigurable structure. In *Proc. of the Conference on Robotics and Automation (ICRA98)*, page 432. IEEE, 1998.

[30] Satoshi Murata, Haruhisa Kurokawa, and Shigeru Kokaji. Self-assembling machine. In *Proc. of the Conference on Robotics and Automation (ICRA94)*, pages 441–448, Los Alamitos, CA, 1994. IEEE.

[31] Amit Pamecha, Imme Ebert-Uphoff, and Gregory S. Chirikjian. Useful metrics for modular robot motion planning. *IEEE Transactions on Robotics and Automation*, 13:531–545, 1997.

[32] D. Rus and M. Vona. Self-reconfiguration planning with compressible unit modules. In *Proc. of the Conference on Robotics and Automation (ICRA99)*. IEEE, 1999.

[33] J. R. Rush, A. P. Fraser, and D. P. Barnes. Evolving cooperation in autonomous robotic systems. In *Proceedings of the IEEE International Conference on Control*, March 21-24 1994.

[34] B. Salemi, W.-M. Shen, and P. Will. Hormone controlled metamorphic robots. In *Proc. of the Intl. Conf. on Robotics and Automation (ICRA2001)*, 2001.

[35] Luc Steels. Cooperation between distributed agents through self-organization. *Journal on Robotics and Autonomous Systems*, 1989.

[36] Darcy Wentworth Thompson. *On Growth and Form*. Cambridge University Press, Cambridge, 1992.

[37] Mark Yim. *Locomotion with a Unit-Modular Reconfigurable Robot*. PhD thesis, Stanford University, 1994.

[38] Mark Yim, John Lamping, Eric Mao, and J. Geoffrey Chase. Rhombic dodecahedron shape for self-assembling robots. Technical Report P97-10777, Xerox PARC, 1997.

ROBOTICS AND PERCEPTION

VISION

Learning Iterative Image Reconstruction

Sven Behnke
Freie Universität Berlin, Institut für Informatik
Takustr. 9, 14195 Berlin, Germany, behnke@inf.fu-berlin.de

Abstract

Successful image reconstruction requires the recognition of a scene and the generation of a clean image of that scene. We propose to use recurrent neural networks for both analysis and synthesis.

The networks have a hierarchical architecture that represents images in multiple scales with different degrees of abstraction. The mapping between these representations is mediated by a local connection structure. We supply the networks with degraded images and train them to reconstruct the originals iteratively. This iterative reconstruction makes it possible to use partial results as context information to resolve ambiguities.

We demonstrate the power of the approach using three examples: superresolution, fill in of occluded parts, and noise removal / contrast enhancement.

1 Introduction

The quality of captured real world images is frequently not sufficient for the application at hand. The reasons for this can be found in the image formation process (e.g. occlusions) and in the capturing device (e.g. low resolution, sensor noise).

Goal of the reconstruction process is to improve the quality of measured images, e.g. by suppressing the noise. To separate noise from objects, models of the noise and the objects present in the images are needed. Then, the scene can be recognized and a clean image of that scene can be generated.

Hierarchical image decompositions using wavelets have been successfully applied to image denoising [Simoncelli and Adelson, 1996; Donoho and Johnstone, 1995]. The image is transformed into a multiscale representation and the statistics of the coefficients of this representation are used to threshold them. The back-projected images are then less noisy. Problematic with these approaches is that the choice of the wavelet transformation is usually fixed and the thresholding ignores dependencies between neighboring locations within a scale and between scales.

The recently proposed VISTA approach [Freeman and Pasztor, 1999] to learning low-level vision uses Markov random fields to model images and scenes. The parameters of these graphical models can be trained, e.g. for a superresolution task. However, the models have no hidden variables and the inference via belief propagation is only approximate.

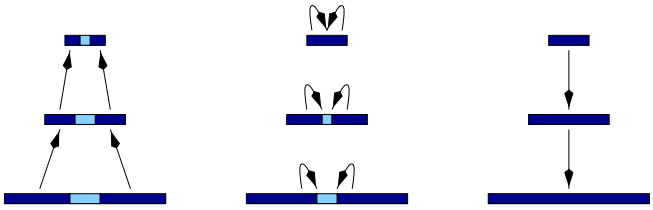


Figure 1: Iterative image reconstruction.

Continuous attractor networks have been proposed to complete images with occlusions [Seung, 1998]. For digits belonging to a common class, a two-layer recurrent network was trained using gradient descent to reconstruct the original. The network had many adaptable parameters, since no weight sharing was used. Further, it was not demonstrated that the reconstruction is possible, if the digit class is unknown. We extend the approach by adding lateral connections, weight sharing, and more layers to the network and train it to reconstruct digits from all classes without presenting the class label.

A common problem with image reconstruction is that it is difficult to decide locally about the interpretation of an image part. For example in a digit binarization task, it might be impossible to decide whether or not a pixel belongs to the foreground by looking only at the pixel's intensity. If contrast is low and noise is present, it could be necessary to bias this decision with the output of a line-detector for that location.

In general, to resolve such local ambiguities, a large context is needed, but feed-forward models that consider such a large context have many free parameters. They are therefore expensive to compute and difficult to train.

We propose to iteratively transform the image into a hierarchical representation and to use partial results as context. Figure 1 illustrates the propagation of information from regions that are interpreted easily to ambiguous regions. Further, we describe the reconstruction problem using examples of degraded images and desired output images and train a recurrent neural network of suitable structure to do the job.

The remainder of the paper is organized as follows: In the next section, the hierarchical architecture of the proposed recurrent networks is introduced. Section 3 discusses the supervised training of such networks. Experiments on three image reconstruction tasks are presented in Section 4.

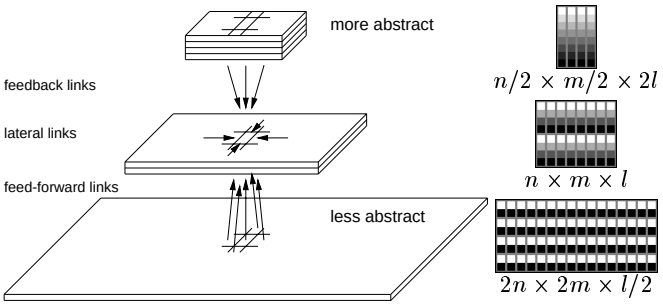


Figure 2: Sketch of the recurrent network.

2 Hierarchical Architecture

The neural abstraction pyramid architecture, introduced in [Behnke and Rojas, 1998], is a suitable framework for iterative image reconstruction.

The main features of the architecture are:

- *Pyramidal shape:* Layers of *columns* are arranged vertically to form a pyramid (see Fig. 2). Each column consists of a set of neural processing elements (*nodes*) with overlapping receptive fields. The number of nodes per column increases and the number of columns per layer decreases towards the top of the pyramid.
- *Analog representation:* Each layer describes an image in a two-dimensional representation where the level of abstraction increases with height, while spatial resolution decreases. The bottom layer stores the given image (a signal). Subsymbolic representations are present in intermediate layers, while the highest layers contain almost symbolic descriptions of the image content. These representations consist of *quantities* that have an *activity value* from a finite interval for each column.
- *Local interaction:* Each node is connected to some nodes from its neighborhood via directed *weighted links*. The shared weights of all nodes in a layer that represent the same quantity are described by a common *template*. The links can be classified as:
  - *feed-forward links:* perform feature extraction,
  - *lateral links:* for consistent interpretation,
  - *feedback links:* provide interpretation hypotheses.
- *Discrete time computation:* The update of a node's value for time step t depends only on the input values at $(t-1)$. All nodes are updated in parallel at each time step.

We use Σ-units as neural processing elements that compute the weighted sum of their inputs and apply a nonlinear output function. The update of the value $v_{x,y,z,q}$ of a unit at column (x, y) in layer z for quantity q is done as follows:

$$v^{t+1}_{x,y,z,q} = \sigma\left[\sum_{j\in\mathcal{L}(i)} \mathcal{W}(j)\, v^{t}_{\mathcal{X}(j,x),\mathcal{Y}(j,y),\mathcal{Z}(j,z),\mathcal{Q}(j)} + \mathcal{B}(i)\right].$$

The template $i = \mathcal{T}(z, q)$ is associated with quantity q at layer z. $\mathcal{L}(i)$ is the set of links of that template and $\mathcal{B}(i)$ is the template bias. $(\mathcal{X}(j, x), \mathcal{Y}(j, y), \mathcal{Z}(j, z), \mathcal{Q}(j))$ describe location and quantity of the input value for link j, and $\mathcal{W}(j)$ is the link weight. The output function $\sigma(x) = 1/(1 + e^{-x})$ is here a sigmoid function that limits the values to the interval $[0, 1]$. In addition to the weights and the bias a start value $\mathcal{V}^0(i)$ for initialization at $t = 0$ is needed for each template. The value of input nodes is set to a copy of the corresponding component of the input vector $\mathbf{x}_k$ of the current example k:

$$v^t_{x,y,z,q} = x^t_{k,\mathcal{I}(x,y,z,q)}, \text{ if } i = \mathcal{T}(z,q) \text{ is input template.}$$

The feed-forward inputs of a node come from all quantities in a small window at the corresponding position in the layer $(z - 1)$ directly below that node. Lateral connections link to all quantities in its neighborhood, including the node itself. Feedback links originate from the units in the layer above that correspond to the same position.

3 Training Recurrent Networks

In [Behnke, 1999] an unsupervised learning algorithm for the neural abstraction pyramid architecture has been proposed. It learns a hierarchy of increasingly abstract representations of the image content that could be used to improve the quality of the images. Here, we apply supervised training to achieve the desired image reconstruction.

Training of recurrent neural networks is difficult due to the non-linear dynamics of the system. Several supervised training methods have been proposed in the literature. Real-time recurrent learning (RTRL) [Williams and Zipser, 1989] is suitable for continuously running networks, but very resource intensive. The backpropagation through time algorithm (BPTT) [Williams and Peng, 1990] unfolds the network in time and applies the backpropagation idea to compute the gradient of the error function. Its computational costs are linear in the number of time steps the error is propagated back.

For image reconstruction, we present a static input $\mathbf{x}_k$ to the network and train it to quickly reach a fixed point that coincides with the desired output $\mathbf{y}_k$. Thus, the network runs only a few iterations and the gradient can be computed efficiently. No artificial truncation of history is necessary.

3.1 Objective Function

The goal of the training is to produce the desired output $\mathbf{y}_k$ as quickly as possible. To achieve this, the network is updated for a fixed number T of iterations. The output vector $\mathbf{v}^t_k$ collects the output units of the network in an appropriate order.

The output error δ^t_k, the difference between the activity of the output units $\mathbf{v}^t_k$ and the desired output $\mathbf{y}_k$ is not only computed at the end of the sequence, but after every update step. In the error function we weight the squared differences progressively, as the number of iterations increases:

$$E = \sum_{k=1}^{K}\sum_{t=1}^{T} t^2 \|\mathbf{y}_k - \mathbf{v}^t_k\|^2.$$

A quadratic weight t^2 has proven to give the later differences a large enough advantage over the earlier differences, such that the network prefers a longer approximation phase, if the final approximation to the desired output is closer.

The contribution of intermediate output values to the error function makes a slight modification to the original backpropagation rule necessary. At all copies of the output units $v^t_{x,y,z,q}$ for $t < T$ the difference $\delta^t_{k,I(x,y,z,q)}$ is computed and added to the backpropagated component of the gradient.

3.2 Robust Gradient Descent

Minimizing the error function with gradient descent faces the problem that the gradient in recurrent networks either vanishes or grows exponentially in time, depending on the magnitude of gains in loops [Bengio *et al.*, 1994]. It is therefore very difficult to determine a learning constant that allows for both stability and fast convergence.

For that reason, we decided to employ the RPROP algorithm [Riedmiller and Braun, 1993], that maintains a learning constant for each weight and uses only the sign of the gradient to determine the weight change. The learning rates are initialized to a moderate value, increased when consecutive steps have the same direction, and decreased otherwise. We modify not only the weights in this way, but adapt the bias and start values as well.

The RPROP training method proved experimentally to be much more stable than gradient descent with a fixed learning rate. However, to compute the gradient, all training examples have to be presented to the network, which is slow for large training sets. To accelerate the training we implemented the following modification. We use as batch only a small working set of training examples. This set is initialized at random. After each weight update, a small fraction of the examples is replaced with randomly chosen examples to ensure a stable estimate for the gradient that takes over time all training examples into account. With a working set of 1% of 60.000 training examples we realized a speedup of two orders of magnitude, as compared to the batch method, without compromising convergence.

4 Experimental Results

We conducted a series of experiments with images of handwritten digits to demonstrate the power of the proposed approach for iterative image reconstruction. The reason for choosing digits was that large datasets are publicly available and that the images contain multiscale structure which can be exploited by the learning algorithm. Clearly, if there were no structure to learn, the training would not help.

We degraded the digits by subsampling, occlusion, or noise and trained recurrent networks to reconstruct the originals.

4.1 Superresolution

For our first experiment we used the original NIST images of segmented binarized handwritten digits [Garris and Wilkinson, 1992]. The digits are given in a 128×128 window, but their bounding box is typically much smaller. For this reason, we centered the bounding box in a 64×64 window to produce the desired output Y. The input X to the network consists of 16×16 subsampled versions of the digits that have been produced by averaging 4×4 pixels.

The superresolution network has three layers, as shown in Figure 3. The low resolution image is input to the rightmost layer. Four 32×32 quantities represent the digit in the middle layer. They are connected to their 3×3-neighborhoods, to 2×2 windows of the output units, and to a single input node. The leftmost layer contains only the output units of the network. They are connected to four nodes in the middle layer and to their 3×3-neighborhoods.

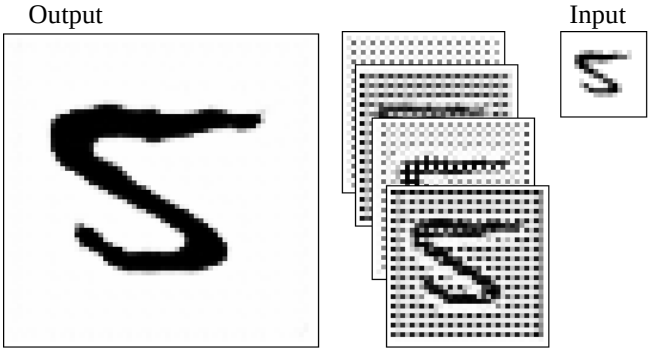


Figure 3: Network for superresolution.

We initialized the 235 free parameters of the network randomly and trained the network for ten time steps using 200 randomly chosen examples. As test set we used 200 different randomly chosen examples. Figure 4 shows for the first five test digits, how the output of the network develops over time. After two iterations the input can influence the output, but no further interactions are possible yet. In the following iterations the initial reconstruction is refined.

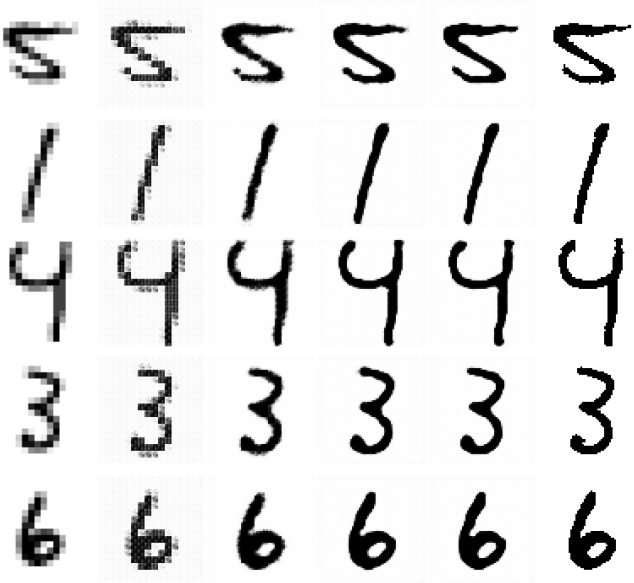


Figure 4: Iterative superresolution.

The network tries to concentrate the gray that is present in the input images at black lines with smooth borders. To illustrate this behavior, we presented uniform pixel noise to the network. The stable response after ten time steps is shown

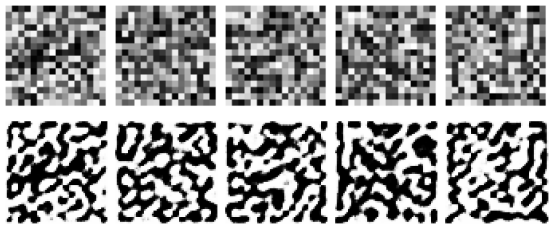


Figure 5: Response of the superresolution network to uniform noise.

in Figure 5. The network hallucinates smooth black lines at positions where many dark pixels are present.

We also trained a larger version of the recurrent network (RNN), that had eight hidden quantities in the middle layer as well as two feed forward neural networks (FFNN) with four and eight quantities. The units of the FFNNs looked at 3×3 windows of the previous layer such that the networks had a similar number of adjustable parameters as the corresponding RNNs. Figure 6 shows for the next five test digits the output of these four networks after 10 iterations. In general, the reconstructions are good approximations to the high resolution targets, given the low resolution inputs. The RNN outputs appear to be sharper than the responses of the FFNNs.

In Figure 7 the mean square error of the networks is displayed. The test set reconstruction error of the recurrent networks decreases quickly and remains below the error of the corresponding FFNN after six time steps. At iterations 9 and 10 the small RNN outperforms even the large FFNN.

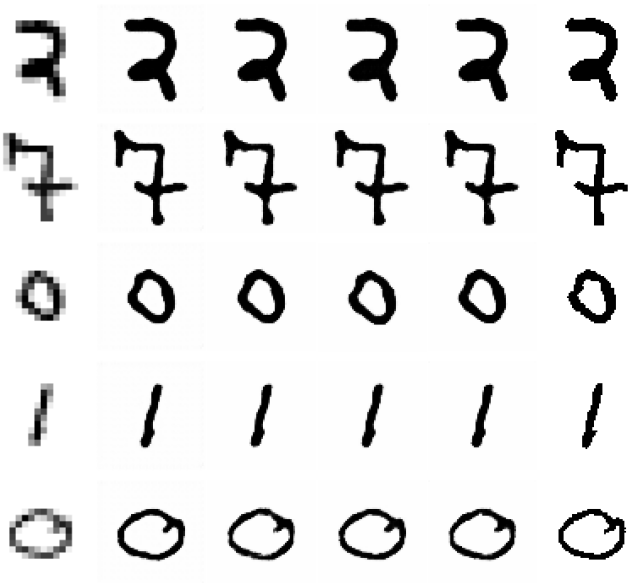


Figure 6: Outputs of different superresolution networks.

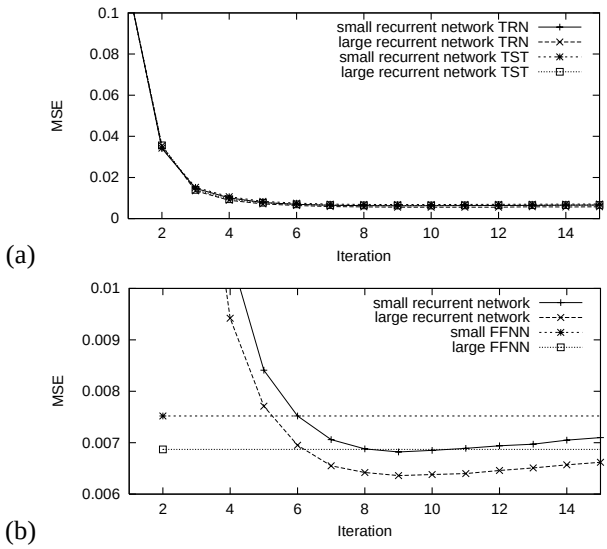


Figure 7: Mean square error of superresolution: (a) the recurrent network on the training set and the test set; (b) detailed view of the test set performance, compared to FFNN.

4.2 Fill In of Occluded Parts

For the second reconstruction experiment we used the MNIST database of handwritten digits [LeCun, 1994]. The NIST digits have been scaled to the size 20×20 and centered in an 28×28 image. We set an 8×8 square to the value 0.125 (light gray) to simulate an occlusion. The square was placed randomly at one of 12×12 positions, leaving a 4 pixel wide border that was never modified.

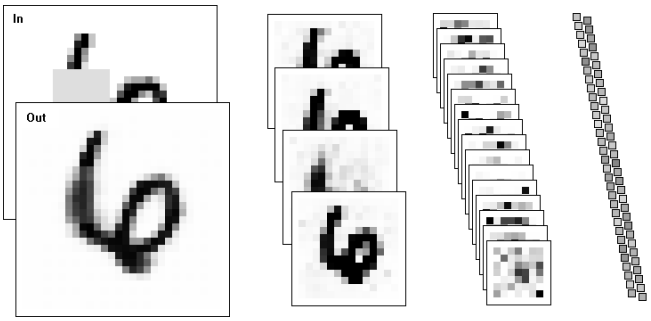


Figure 8: Network for fill in of occluded parts.

The reconstruction network consisted of four layers, as illustrated in Figure 8. The first layer (28×28) contains the input image and the output units of the network. In the second layer four quantities with resolution 14×14 look at overlapping 4×4 windows of the quantities below. The 16 quantities in the third layer have also 4×4 feed-forward connections, while in the top layer the resolution of the 64 quantities is reduced to 1×1 and the feed-forward weights are connected to all $7 \times 7 \times 16$ nodes of the third layer. The first three layers are surrounded by a one pixel wide border that is set to zero. In these layers the nodes have 3×3 lateral connections. In the fourth layer the lateral weights contact all 64 nodes. The feed-

back links are non-overlapping and have thus the size 2×2 between the first three layers and 7×7 between the third and the topmost layer.

Training is done with a working set of 600 of the 60.000 examples for twelve time steps. Figure 9 displays the reconstruction process for the first ten digits of the test set. One can observe that the images change mostly at occluded pixels. This shows that the network recognized the occluding square. Further, the change is such that a reasonable guess is produced, how the digit could look like behind the square. The network connects lines again that have been interrupted by the square. It is also able to extend shortened lines and to close opened loops. In most cases, the reconstructions are very similar to the original digits.

4.3 Noise Removal and Contrast Enhancement

The last experiment uses the same network architecture and the same MNIST digits, but degrades the input images as follows. We scaled the pixel intensities to $[0.25, 0.75]$, added a random background level that was uniformly distributed in the range $(-0.25, 0.25)$, and added uniform pixel noise in the range $(-0.25, 0.25)$. Finally, we clipped the pixel values at $[0, 1]$. The first column of Figure 10 shows the first ten digits of the test set that have been corrupted in this way. The network was trained on a working set of 600 out of 60.000 digits for twelve time steps.

The reconstruction process is also shown in Figure 10. One can observe that the network is able to detect the dark lines, to complete them, to remove the background clutter, and to enhance the contrast. The interpretation of most locations is decided quickly by the network. Ambiguous locations are kept for some iterations at intermediate values, such that the decision can be influenced by neighboring nodes. The reconstructed digits are very similar to the originals.

5 Discussion

The experiments demonstrated that difficult non-linear image reconstruction tasks can be learned by hierarchical neural networks with local connectivity. Supervised training of the networks was done by a combination of BPTT and RPROP.

The networks reconstruct images iteratively and are able to integrate partial results as context information for the resolution of local ambiguities. This is similar to the recently demonstrated belief propagation in graphical networks with cycles. The difference is that the proposed approach learns horizontal and vertical feedback loops that produce rich multiscale representations to model the images where current belief propagation approaches use either trees or arrays to represent the vertical or horizontal dependencies, respectively.

Further, the proposed network can be trained to compute an objective function directly, while inference in belief networks with cycles is only approximate due to multiple counting of the same evidence.

Recently, generalized belief propagation has been proposed [Yedidia *et al.*, 2001] that allows for better approximations of the inference process. It would be interesting to investigate the relationship between this approach and the proposed hierarchical recurrent neural networks.

The iterative reconstruction is not restricted to static images. The training method allows for a change of input and/or desired output at each time step. Thus, the networks should be able to integrate information over time, which would help to reconstruct video sequences.

References

[Behnke and Rojas, 1998] Sven Behnke and Raúl Rojas. Neural abstraction pyramid: A hierarchical image understanding architecture. In *Proceedings IJCNN'98–Anchorage*, volume 2, pages 820–825, 1998.

[Behnke, 1999] Sven Behnke. Hebbian learning and competition in the neural abstraction pyramid. In *Proceedings IJCNN'99 – Washington, DC, paper #491*, 1999.

[Bengio *et al.*, 1994] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[Donoho and Johnstone, 1995] D.L. Donoho and I.M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90(432):1200–1224, 1995.

[Freeman and Pasztor, 1999] W. Freeman and E. Pasztor. Learning low-level vision. In *Proceedings of ICCV99*, pages 1182–1189, 1999.

[Garris and Wilkinson, 1992] M. D. Garris and R. A. Wilkinson. NIST special database 3 – handwritten segmented characters. Technical Report HWSC, NIST, 1992.

[LeCun, 1994] Yann LeCun. The MNIST database of handwritten digits. http://www.research.att.com/~yann/exdb/mnist, AT&T Labs, 1994.

[Riedmiller and Braun, 1993] Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proceedings of the International Conference on Neural Networks – San Francisco, CA*, pages 586–591. IEEE, 1993.

[Seung, 1998] H. Sebastian Seung. Learning continuous attractors in recurrent networks. In *Advances in Neural Information Processing Systems 10*, pages 654–660, 1998.

[Simoncelli and Adelson, 1996] E. Simoncelli and E. Adelson. Noise removal via Bayesian wavelet coring. In *Proceedings of IEEE Int. Conf. on Image Processing — ICIP'96 (Switzerland)*, 1996.

[Williams and Peng, 1990] R. Williams and J. Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):491–501, 1990.

[Williams and Zipser, 1989] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[Yedidia *et al.*, 2001] J. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems 13 (to appear)*, 2001.

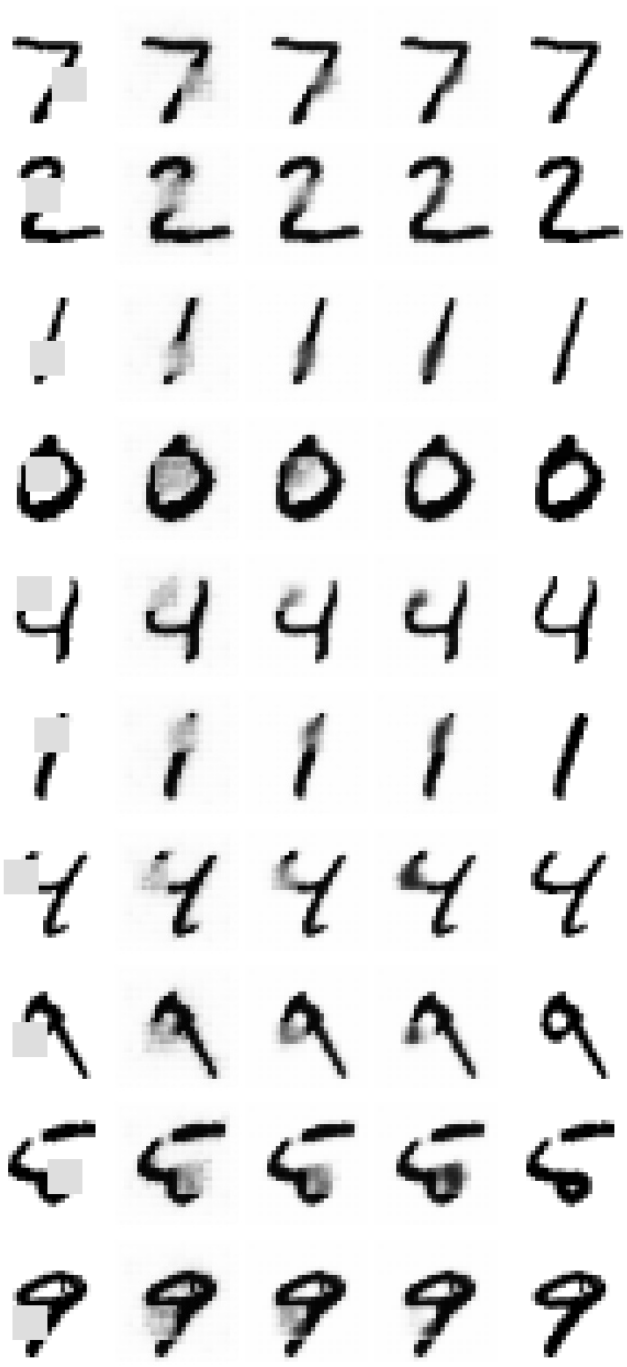


Figure 9: Fill in of occluded parts.

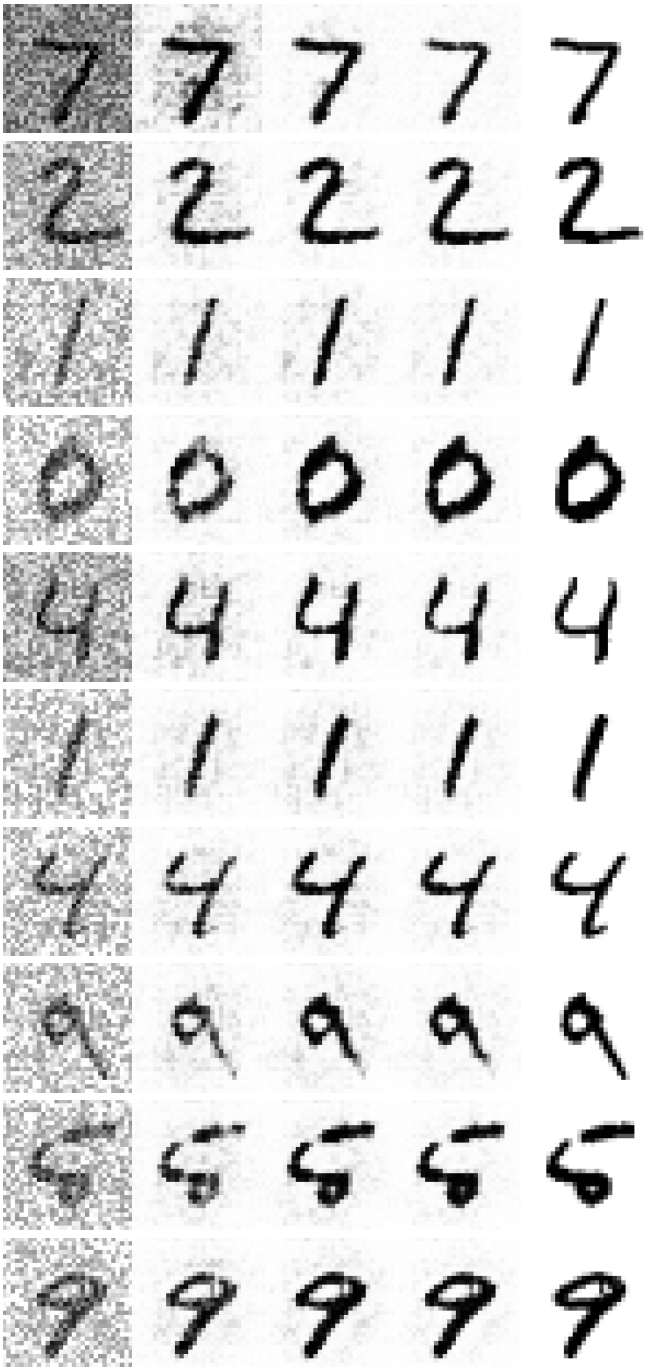


Figure 10: Noise removal and contrast enhancement.

A Hierarchy of Boundary-based Shape Descriptors

Richard Meathrel and **Antony Galton**
School of Engineering and Computer Science
University of Exeter
Exeter EX4 4PT, UK
E-mail: {R.C.Meathrel, A.P.Galton}@ex.ac.uk

Abstract

In this paper we extend previous work on the boundary-based approach to describing shape, by deriving an unbounded hierarchy of "atomic" shape descriptors (called tokens) based on tangent bearing and its successive derivatives, and incorporating angle and cusp curve features. Both open and closed curves have token-string descriptions at all levels in the hierarchy. We provide a pair of compatibility matrices for generating transition tables for any level, from which level-specific token ordering graphs that encode basic string syntax can be systematically constructed.

1 Introduction

In the development of conceptual tools for spatial representation and reasoning, the category of shape has proved to be one of the most problematic areas. Some important earlier work, representative of the boundary-based approach to shape, is exemplified by the contour codons of Hoffman and Richards [1982] and the extremum primitives of Leyton [1988]. These two approaches, although differing in motivation and detail, both used the idea of characterising the shape of an outline by means of a string of tokens, recording salient curvature-based features encountered during a traversal of the outline. A more recent approach, using yet another set of primitives, is that of [Galton and Meathrel, 1999]. In [Meathrel and Galton, 2000], we attempted to define the boundary-based approach to shape in a more general and systematic way. By considering variations in curvature as a starting point, we derived two sets of *atomic tokens* for describing curves, and presented *token ordering graphs* for verifying the syntax of atomic curve descriptions.

The present paper extends the work described in [Meathrel and Galton, 2000] in three significant respects: (i) by systematically investigating the structure of sets of atomic curvature-based tokens, resulting in an unbounded hierarchy of atomic tokens, (ii) by incorporating kink points (angles and cusps) into the framework of tokens, and (iii) by providing a pair of *compatibility matrices*, and an algorithm, for generating transition tables for any level in the hierarchy, from which level-specific token ordering graphs can be constructed.

2 Deriving the hierarchy of atomic tokens

At each point on a curve, the tangent to the curve at the point is either defined or undefined. The rate of change of the tangent bearing b with distance along a curve gives us curvature (the first derivative of tangent bearing with respect to arc-length). We can also consider the rate of change of curvature (c'), which is the second derivative of tangent bearing. For each curve we can think of there being an infinite number of associated plots. Our hierarchy of descriptors is based on a discretisation of b and its derivatives. For tangent bearing we are interested in whether b is defined (D) or undefined (U), so we use the quantity space $\{D, U\}$. Points where the tangent bearing is undefined correspond to angles and cusps, and may be referred to as *kink points*. For all of the derivatives of tangent bearing (i.e., $c, c', c'', \ldots$), we use the quantity space $\{+, 0, -, U\}$, since a derivative may be positive, zero, negative, or undefined.

2.1 Curve states

Associated with each point P on a curve is a sequence of qualitative values, representing b, c, c', etc. The *complete curve state* at P corresponds to an infinite sequence of values. We write $\partial^k x$ to denote the kth component of the curve state x. Not all component sequences give rise to valid curve states. If a component has a value other than U, then the value of the next component is unconstrained. If, however, a component *does* have the value U, then all subsequent components must also be U. More formally, a curve state x must satisfy the following constraint:

$$(\forall k)(\, \partial^k x = U \rightarrow \partial^{k+1} x = U \,)$$

A *partial curve state* at P is any initial n-tuple of the complete curve state. So if $x = \langle D, + \rangle$, for example, then $\partial^1 x = D$, $\partial^2 x = +$, and x is a partial curve state that is assigned to points on a curve where the tangent bearing is defined and the curvature is positive.

We refer to a partial curve state with n components as a level-n state. Given the complete set of states for some level k, it is straightforward to generate the set of states for level $k{+}1$. Each state x, at level k, generates a set of level-$k{+}1$ states, $\mathcal{S}$, as follows (where $y = \partial^k x$):

$$\mathcal{S} = \begin{cases} \{\, \langle \ldots, y, U \rangle \,\} & \text{if } y = U \\ \{\, \langle \ldots, y, + \rangle,\ \langle \ldots, y, 0 \rangle, & \text{otherwise} \\ \qquad \langle \ldots, y, - \rangle,\ \langle \ldots, y, U \rangle \,\} & \end{cases}$$

There are two partial curve states with one component: $\langle D \rangle$ and $\langle U \rangle$. From these two states we can generate the five states of level 2 and then, from those, the fourteen states of level 3:

$\langle D \rangle$: $\langle D,+\rangle$, $\langle D,0\rangle$, $\langle D,-\rangle$, $\langle D,U\rangle$
$\langle U \rangle$: $\langle U,U\rangle$

$\langle D,+\rangle$: $\langle D,+,+\rangle$, $\langle D,+,0\rangle$, $\langle D,+,-\rangle$, $\langle D,+,U\rangle$
$\langle D,0\rangle$: $\langle D,0,+\rangle$, $\langle D,0,0\rangle$, $\langle D,0,-\rangle$, $\langle D,0,U\rangle$
$\langle D,-\rangle$: $\langle D,-,+\rangle$, $\langle D,-,0\rangle$, $\langle D,-,-\rangle$, $\langle D,-,U\rangle$
$\langle D,U\rangle$: $\langle D,U,U\rangle$
$\langle U,U\rangle$: $\langle U,U,U\rangle$

For each point on a curve we can assign a partial curve state of k components, so we could theoretically represent a curve by providing a mapping between curve points and partial curve states. However, such a mapping would be infinite and therefore of no practical use. Because our state components take qualitative values, however, certain states may persist over intervals of curve and, therefore, support mappings that are finite. A curve state may have an interval *and/or* a point interpretation. A state that has an interval interpretation may persist over an interval of curve, and a state that has a point interpretation may hold at a single curve point, without holding on any interval adjoining that point. An *atomic token* identifies a particular interpretation of a partial curve state.

2.2 Interval and point interpretation

For each set of level-k curve states, we obtain a corresponding set of level-k atomic tokens by considering the allowable interpretations of each state. A state may support an interval interpretation, a point interpretation, or both. An atomic token (or "atom") is a particular interpretation of a particular state, and is identified by a *signature* consisting of a sequence of qualitative component values. A signature that is underlined indicates an interval interpretation; a non-underlined signature indicates a point interpretation. The atom D+, for example, is identified with the interval interpretation of the state $\langle D,+\rangle$, and the atom D+-0 is identified with the point interpretation of the state $\langle D,+,-,0\rangle$.

A curve state may hold at a single point *iff* one of its components is either zero or undefined. This is because if all components are defined, then all of them are continuous, and hence the values '$+$' and '$-$' can only hold over intervals. A curve state may persist over an interval *iff* none of its components are undefined and, whenever a component has the value zero, the next component also has the value zero. The following predicates, therefore, can be used to determine the interpretations that are supported by a curve state:

$$\mathsf{point\text{-}interp}(x) \leftrightarrow (\exists k)(\partial^k x = U \vee \partial^k x = 0)$$
$$\mathsf{interval\text{-}interp}(x) \leftrightarrow (\neg\exists k)(\partial^k x = U) \wedge (\forall k)(\partial^k x = 0 \rightarrow \partial^{k+1} x = 0)$$

2.3 Atomic hierarchy

For each set of partial curve states (each level) we can use the predicates point-interp and interval-interp to obtain the corresponding set of atoms. The partial curve states and atoms for level 3 are as follows:

$\langle D,+,+\rangle$: D++
$\langle D,+,0\rangle$: D+0, D+0
$\langle D,+,-\rangle$: D+-
$\langle D,+,U\rangle$: D+U
$\langle D,0,+\rangle$: D0+
$\langle D,0,0\rangle$: D00, D00
$\langle D,0,-\rangle$: D0-
$\langle D,0,U\rangle$: D0U
$\langle D,-,+\rangle$: D-+
$\langle D,-,0\rangle$: D-0, D-0
$\langle D,-,-\rangle$: D--
$\langle D,-,U\rangle$: D-U
$\langle D,U,U\rangle$: DUU
$\langle U,U,U\rangle$: UUU

The first four levels of the hierarchy of atomic tokens are shown in Figure 1. Each atom in the hierarchy (except the atoms D and U at level 1) is a child of a single parent atom at the previous level, i.e., each atom at level k is derivable from one, and only one, atom at level $k-1$. We call parent atoms that have more than one child *expansive*, and those with only one child *non-expansive*. An interval atom is expansive *iff* its last component is either '$+$' or '$-$', otherwise it is non-expansive. A point atom is expansive *iff* its last component is not 'U', otherwise it is non-expansive. At level 1, then, there is just one non-expansive atom, U, which propagates through the hierarchy as UU, UUU, and so on. The other level-1 atom, D, expands into five atoms at level 2; two of which are non-expansive: D0 and DU.

Any atom is either an interval (I) or a point (P), and either expansive (E) or non-expansive (N), giving us four kinds of atoms: IE, IN, PE, and PN. Each IE atom yields five children: two IE atoms, one IN atom, one PE atom, and one PN atom. Each PE atom yields four children: three PE atoms and one PN atom. The non-expansive atoms, IN and PN, yield a single IN atom and a single PN atom, respectively.

The recursive equations for calculating the numbers of each kind of atom at level k, and the actual figures for levels 1 to 6, are as follows:

$$\begin{aligned} \mathrm{IE}(k) &= 2 \times \mathrm{IE}(k-1) \\ \mathrm{IN}(k) &= \mathrm{IN}(k-1) + \mathrm{IE}(k-1) \\ \mathrm{PE}(k) &= 3 \times \mathrm{PE}(k-1) + \mathrm{IE}(k-1) \\ \mathrm{PN}(k) &= \mathrm{PN}(k-1) + \mathrm{PE}(k-1) + \mathrm{IE}(k-1) \\ \mathrm{Total}(k) &= \mathrm{IE}(k) + \mathrm{IN}(k) + \mathrm{PE}(k) + \mathrm{PN}(k) \end{aligned}$$

Level	IE	IN	PE	PN	Total
1	1	0	0	1	2
2	2	1	1	2	6
3	4	3	5	5	17
4	8	7	19	14	48
5	16	15	65	41	137
6	32	31	211	122	396

2.4 Atom labelling

Annotating curves with signatures and describing curves with strings of signatures is not ideal. Therefore, for notational convenience, we assign each atom at level two or above a suitable descriptive label, according to the following convention[1]: the label begins with 'P', 'Z', 'N', or 'U', depending

[1] At level 1, D is given the label 'D' and U is given the label 'U_b', where 'b' signifies that the tangent bearing component is undefined.

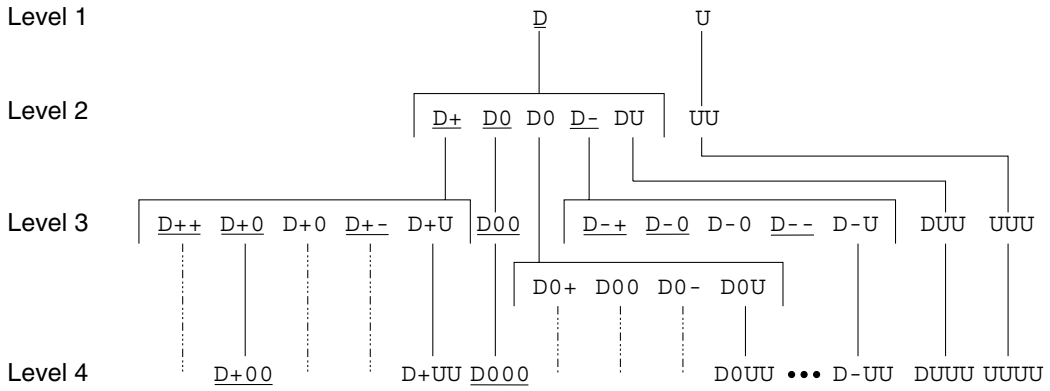


Figure 1: The first four levels of the atomic hierarchy

on whether the curvature component is positive, zero, negative, or undefined. The remaining components ($c', c'', \ldots$) are denoted by a superscripted string, whose elements are values from the set $\{+, 0, -, U\}$. The label is underlined if the atom it is symbolising is an interval. To distinguish between the two distinct kinds of atoms where the curvature component is U, we use the label '$\mathsf{U_b}$' for atoms where the tangent bearing component is also U, and '$\mathsf{U_c}$' for atoms where the tangent bearing is D. Using the new notation, the first three sets of atomic tokens are written as follows:

Level 1 : $\underline{\mathsf{D}}, \mathsf{U_b}$

Level 2 : $\underline{\mathsf{P}}, \underline{\mathsf{Z}}, \mathsf{Z}, \underline{\mathsf{N}}, \mathsf{U_c}, \mathsf{U_b}$

Level 3 : $\underline{\mathsf{P}^+}, \underline{\mathsf{P}^0}, \mathsf{P}^0, \underline{\mathsf{P}^-}, \mathsf{P}^U, \mathsf{Z}^+, \underline{\mathsf{Z}^0}, \mathsf{Z}^0, \mathsf{Z}^-, \mathsf{Z}^U, \underline{\mathsf{N}^+}, \underline{\mathsf{N}^0}, \mathsf{N}^0, \underline{\mathsf{N}^-}, \mathsf{N}^U, \mathsf{U_c}, \mathsf{U_b}$

2.5 Kink points

At each level in the hierarchy there exists a single kink-point atom labelled $\mathsf{U_b}$, each component of which takes the value U. Such atoms represent points on a curve where the tangent bearing (and therefore all of its derivatives) are undefined, and correspond perceptually to angles and cusps. Clearly, we would like to be able to distinguish between different kinds of kink points, otherwise we are losing important curve information. In particular, the two most relevant aspects of a kink point we would like to preserve are its orientation (whether it is inward-pointing or outward-pointing) and its type, i.e., whether it is an angle or a cusp. In order to distinguish between the different kinds of kink points, we introduce the following *kink tokens* into our representation: $\mathsf{U_<}$ and $\mathsf{U_>}$ (for inward and outward pointing angles, respectively), and $\mathsf{U_\prec}$ and $\mathsf{U_\succ}$ (for inward and outward pointing cusps, respectively). These tokens have the same status as $\mathsf{U_b}$, in that they can be thought of as appearing at every level in the hierarchy. Note, however, that kink tokens are not *atomic*, since the process by which the atomic tokens are derived does not produce them. Instead, we need to introduce them explicitly.

3 Atomic description

A curve is described by a string of atomic tokens taken from a particular level in the hierarchy. By prefixing descriptions with a symbol indicating curve type, both open and closed curves can be represented. We use '$\frown$' for open curves and '$\bigcirc$' for closed ones. Our convention for relating changes in tangent bearing to curvature is that a clockwise change in tangent bearing indicates positive curvature, while an anticlockwise change indicates negative curvature. Thus positive curvature is associated with convex curve segments and negative curvature with concave segments. A curve can be traversed in one of two directions. For a closed curve, the direction is that which preserves the intended figure/ground relationship. For open curves, consistent description dictates that one of the two directions is chosen as the "default" one.

Shown in Figure 2 are two curves that have been annotated with level-3 atoms. Note that $\underline{\mathsf{P}^0}$ refers to a convex circular arc (constant positive curvature over an interval), whereas P^0 refers to a positive curvature extremum; and analogously with $\underline{\mathsf{N}^0}$ and N^0. Given a closed curve there will be, in general, a number of different token strings that describe it. However, we can easily transform one string into another by cyclically shifting it by a certain amount. In this sense, all of the strings may be considered equivalent. Any one of sixteen level-3 strings may be used to describe the closed curve in Figure 2, two of which are obtained by either starting at the $\mathsf{U_<}$ or starting at the uppermost $\underline{\mathsf{Z}^0}$:

$$\bigcirc\, \mathsf{U_<}\, \underline{\mathsf{P}^+}\, \mathsf{P}^0\, \underline{\mathsf{P}^-}\, \mathsf{Z}^-\, \underline{\mathsf{N}^-}\, \mathsf{N}^0\, \underline{\mathsf{N}^+}\, \mathsf{U_\succ}\, \underline{\mathsf{Z}^0}\, \mathsf{U_c}\, \underline{\mathsf{P}^0}\, \mathsf{U_>}\, \underline{\mathsf{Z}^0}\, \mathsf{U_\prec}\, \underline{\mathsf{P}^0}$$
$$\bigcirc\, \underline{\mathsf{Z}^0}\, \mathsf{U_c}\, \underline{\mathsf{P}^0}\, \mathsf{U_>}\, \underline{\mathsf{Z}^0}\, \mathsf{U_\prec}\, \underline{\mathsf{P}^0}\, \mathsf{U_<}\, \underline{\mathsf{P}^+}\, \mathsf{P}^0\, \underline{\mathsf{P}^-}\, \mathsf{Z}^-\, \underline{\mathsf{N}^-}\, \mathsf{N}^0\, \underline{\mathsf{N}^+}\, \mathsf{U_\succ}$$

The importance of specifying a particular direction for open curves is illustrated by the open curve in Figure 2, where a reversal of direction yields the description $\frown\, \underline{\mathsf{Z}^0}\, \mathsf{U_<}\, \underline{\mathsf{N}^-}$. Note, however, that under a system where "mirror-reflected" curves are considered equivalent, the direction chosen for traversal is unimportant.

3.1 Levels of description

A curve has a description at every atomic level. The most coarse-grained description is at level 1, where only the atoms $\underline{\mathsf{D}}$ and $\mathsf{U_b}$ are available. By moving down the hierarchy (and incorporating more qualitative components) we get finer-grained descriptions that reflect the increase in discriminatory power that extra components provide.

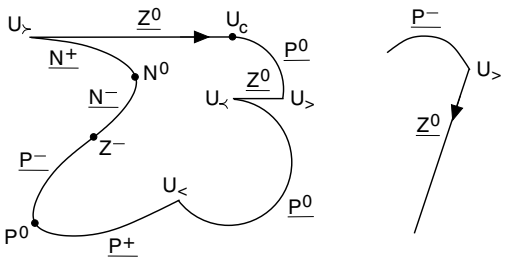


Figure 2: Example curves labelled with level-3 atoms

Given the description of a curve at some level k, we can derive all of the coarser-grained descriptions for the curve, i.e., from the description at level 1 through to the description at level $k-1$. If X denotes the description of a curve $\mathcal{C}$, at level $k > 1$, then the following straightforward three-step procedure yields the description of $\mathcal{C}$ at level $k-1$:

1. Replace each atom of X with its parent.
2. Iteratively replace substrings of the form xx (where x is a single atom) with x, until there are no more substrings of the form xx.
3. If $\mathcal{C}$ is a closed curve, and the string resulting from the previous step is of length greater than two, and begins and ends with atoms of the same type, then remove the last atom in the string.

By applying the procedure to the ellipse shown in Figure 3, we get the level-2 description $\bigcirc\,\underline{\mathsf{P}}$, by re-applying the procedure with $\bigcirc\,\underline{\mathsf{P}}$ as input, we get the most coarse-grained description for an ellipse: $\bigcirc\,\underline{\mathsf{D}}$.

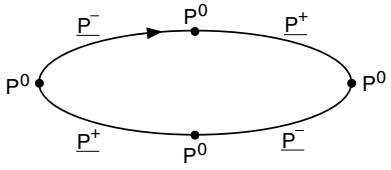


Figure 3: An ellipse described at level 3

Although we can derive all coarser-grained descriptions for a curve given its description at level $k > 1$, it is not possible, in general, to derive any of its finer-grained descriptions. The occasion when it *is* possible to derive a finer-grained description is when the description given consists entirely of atoms that are non-expansive, in which case *all* finer-grained descriptions are accessible. Consider the description of a square at level 2: $\bigcirc\,\underline{\mathsf{Z}}\,\mathsf{U}_{>}\,\underline{\mathsf{Z}}\,\mathsf{U}_{>}\,\underline{\mathsf{Z}}\,\mathsf{U}_{>}\,\underline{\mathsf{Z}}\,\mathsf{U}_{>}$. Because both $\underline{\mathsf{Z}}$ and $\mathsf{U}_{>}$ are non-expansive, we can get the "finer-grained" description at level 3 by replacing each atom in the string with its single child atom. The level-5 description, for example, is $\bigcirc\,\underline{\mathsf{Z}^{000}}\,\mathsf{U}_{>}\,\underline{\mathsf{Z}^{000}}\,\mathsf{U}_{>}\,\underline{\mathsf{Z}^{000}}\,\mathsf{U}_{>}\,\underline{\mathsf{Z}^{000}}\,\mathsf{U}_{>}$. Another example would be a shape that has the level-3 description $\frown\,\underline{\mathsf{P}^{0}}$ (i.e., a convex circular arc segment). For most shapes it would appear that there exists some minimal value of k such that its description at level k contains only non-expansive atoms. For the square, $k = 2$; descriptions at levels $k > 2$ are unnecessary, leading to no increase in "qualitative precision". For a circle, we need to go to level 3 to achieve this. In this sense, a circle might be regarded as a more complex shape than a square.

4 Token ordering graphs

A curve is described by stringing together those atoms that correspond to the "atomic" qualitative boundary features of the curve. Clearly, some features can occur together and some cannot. A token ordering graph (TOG), of the kind introduced in [Meathrel and Galton, 2000], encodes the constraints that determine the basic string syntax for a set of atoms. In this section, we show how the transition tables that underlie such graphs can be systematically constructed for each level of the atomic hierarchy.

Given a set of atoms, a TOG is a way of pictorially representing the two complementary transition tables that specify which atoms may follow which other atoms in a string. The *interval-interval* ("I-I") table tells us which interval atoms can follow which other interval atoms. The *interval-point-interval* ("I-P-I") table tells us which point atoms can occur in between each pair of interval atoms. It is the construction of these tables, then, that we are primarily interested in. The I-I and I-P-I tables for level 2 are given later, in Figure 4.

4.1 Compatibility matrices

We make use of two *compatibility matrices* when constructing I-I and I-P-I tables. First, consider the construction of an I-I table for a given level in the hierarchy. Each cell $(\text{row } x, \text{column } y)$ in an I-I table contains a tick *iff* the interval atom y can directly follow the interval atom x. Given interval atoms x and y, at level n, we can determine whether ot not y can directly follow x by making use of the matrix given in Table 1. The matrix tells us whether or not the kth qualitative components of the atoms x and y are compatible.

	$\partial^k y = +$	$\partial^k y = 0$	$\partial^k y = -$
$\partial^k x = +$	$\top$	$\partial^{k+1} x = -$	$\bot$
$\partial^k x = 0$	$\partial^{k+1} y = +$	$\top$	$\partial^{k+1} y = -$
$\partial^k x = -$	$\bot$	$\partial^{k+1} x = +$	$\top$

Table 1: Compatibility matrix for I-I tables

The kth component of y is compatible with the kth component of x, and $\mathsf{compatible}(k, y, x)$ holds, *iff* the condition in the corresponding cell is satisfied. If $\partial^k x$ and $\partial^k y$ are of opposing signs then they are definitely incompatible (signified by a $\bot$ in the appropriate cell), because a derivative cannot change from positive to negative (or vice versa) without taking the value 0 or U in between. If an interval over which ∂^k has the value 0 is followed immediately by an interval over which ∂^k is positive, then in the latter interval it must initially be increasing, i.e., ∂^{k+1} must be positive. This explains the entry for cell $(\partial^k x = 0, \partial^k y = +)$. Analogous

	$\partial^k y = +$	$\partial^k y = 0$	$\partial^k y = -$
$\partial^k x = +$	$\partial^k p = + \ \vee \ \partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} x = - \ \wedge \partial^{k+1} y = +)$	$\partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} x = -)$	$\partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} x = - \ \wedge \partial^{k+1} y = -)$
$\partial^k x = 0$	$\partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} y = +)$	$\partial^{k-1} p = U$	$\partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} y = -)$
$\partial^k x = -$	$\partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} x = + \ \wedge \partial^{k+1} y = +)$	$\partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} x = +)$	$\partial^k p = - \ \vee \ \partial^k p = U \vee$ $(\partial^k p = 0 \ \wedge \partial^{k+1} x = + \ \wedge \partial^{k+1} y = -)$

Table 2: Compatibility matrix for I-P-I tables

explanations hold for the other non-trivial entries in the table. Given interval atoms x and y at level n, then, y can directly follow x *iff* $\mathsf{can\text{-}follow}(y, x)$ holds:

$$\mathsf{can\text{-}follow}(y, x) \leftrightarrow \\ y \neq x \ \wedge \ (\forall k)(\, 1 < k \leq n \rightarrow \mathsf{compatible}(k, y, x)\,)$$

In other words, y and x must be distinct interval atoms (to ensure an underlying state change), and the components of y must be compatible with the components of x. Note that, whenever a condition in the matrix refers to an "unavailable" component of an atom α, i.e., when $k{+}1 > n$, the condition *is* satisfied because the value of $\partial^{n+1}\alpha$ is unconstrained (since it is not specified). As an example, consider the level-3 interval atoms $\underline{\mathsf{P}^+}$ and $\underline{\mathsf{P}^0}$, and whether $\underline{\mathsf{P}^0}$ can directly follow $\underline{\mathsf{P}^+}$. We start by comparing the *second* component of each atom, since all interval atoms have the same value (D) for their first component. Because both $\partial^2\,\underline{\mathsf{P}^+}$ and $\partial^2\,\underline{\mathsf{P}^0}$ take the value '+', the compatibility condition is $\top$, and therefore satisfied. The condition of compatibility for the final component pair evaluates to $\partial^4\,\underline{\mathsf{P}^+} = -$, since $\partial^3\,\underline{\mathsf{P}^+} = +$ and $\partial^3\,\underline{\mathsf{P}^0} = 0$. The condition is satisfied because $\underline{\mathsf{P}^+}$ has no fourth component ($\underline{\mathsf{P}^+}$ is a generalisation of a set of child atoms that includes the level-4 atom $\underline{\mathsf{P}^{+-}}$).

Next, consider the construction of an I-P-I table for a given level in the hierarchy. Each cell (row x, column y) in an I-P-I table contains a list of point atoms that can occur in between interval atoms x and y. For I-P-I tables, we make use of the compatibility matrix given in Table 2, which differs from the matrix for I-I tables in that the notion of compatibility concerns *three* components rather than just two. Given a point atom p and interval atoms x and y, at level n, the component $\partial^k p$ is compatible with the components $\partial^k x$ and $\partial^k y$, and $\mathsf{compatible}(k, p, x, y)$ holds, *iff* the condition in the corresponding cell is satisfied. A point atom p, then, can occur in between two interval atoms x and y (after x and before y) *iff* $\mathsf{can\text{-}occur\text{-}between}(p, x, y)$ holds:

$$\mathsf{can\text{-}occur\text{-}between}(p, x, y) \leftrightarrow \\ (\exists a, b \leq n)(\, \partial^a p \neq \partial^a x \ \wedge \ \partial^b p \neq \partial^b y\,) \\ \wedge\, (\forall k)(\, 1 < k \leq n \rightarrow \mathsf{compatible}(k, p, x, y)\,)$$

In other words, the underlying curve state of p must be distinct from the underlying curve states of x and y, and the qualitative components of p must be compatible with those of x and y. As before, conditions that refer to "unavailable" components are automatically satisfied. To illustrate we shall use the table to determine which point atoms can come between $x = \underline{\mathsf{P}^-}$ and $y = \underline{\mathsf{P}^0}$. We have $\partial^2 x = \partial^2 y = +$ and $\partial^3 x = -, \partial^3 y = 0$. We want to know the possible values for $\partial^1 p$, $\partial^2 p$ and $\partial^3 p$. Since the definition of $\mathsf{can\text{-}occur\text{-}between}$ imposes no constraints on the first level components, $\partial^1 p$ can be either D or U. For $k = 2$, the top-left cell of the matrix states that $\partial^2 p$ may be any of $+$, U, and 0, but 0 is ruled out because it requires that $\partial^3 x = -$ and $\partial^3 y = +$, which contradicts $\partial^3 y = 0$. Moving on to $k = 3$, we consult the middle cell in the bottom row of the matrix. We see that $\partial^3 p$ may be either U or 0, the additional constraints on the latter value being irrelevant in the present case, as $k{+}1 > 3$. Remembering that if any component is U, all subsequent components must be U too, we derive the following candidate component sequences for p:

$$\langle U, U, U\rangle, \langle D, +, U\rangle, \langle D, +, 0\rangle, \langle D, U, U\rangle$$

corresponding to the level-3 atoms $\mathsf{U_b}$, $\mathsf{P^U}$, $\mathsf{P^0}$, and $\mathsf{U_c}$. The first condition of $\mathsf{can\text{-}occur\text{-}between}$ rules out $\mathsf{P^0}$, as its underlying curve state is not distinct from that of y. Thus we conclude that only the point atoms $\mathsf{U_b}$, $\mathsf{P^U}$, and $\mathsf{U_c}$ can occur in between the interval atoms $\underline{\mathsf{P}^-}$ and $\underline{\mathsf{P}^0}$.

4.2 Transition tables

The set of atoms at level k is the union of a set of interval atoms, $\mathcal{I}_k$, and a set of point atoms, $\mathcal{P}_k$, e.g., at level 2 we have $\mathcal{I}_2 = \{\underline{\mathsf{P}}, \underline{\mathsf{Z}}, \underline{\mathsf{N}}\}$ and $\mathcal{P}_2 = \{\mathsf{Z}, \mathsf{U_c}, \mathsf{U_b}\}$. The transition tables at level k, I-I_k and I-P-I_k, each contain $|\mathcal{I}_k|^2$ cells, one for each ordered pair of interval atoms. We refer to a cell (row x, column y) in the I-I_k and I-P-I_k tables by writing I-$\mathrm{I}_k(x, y)$ and I-P-$\mathrm{I}_k(x, y)$, respectively. The following algorithm populates the I-I and I-P-I tables for any level $k > 1$ in the hierarchy[2]:

[2] At level 1, I-$\mathrm{I}_1(\underline{\mathsf{D}}, \underline{\mathsf{D}})$ is empty and I-P-$\mathrm{I}_1(\underline{\mathsf{D}}, \underline{\mathsf{D}})$ contains $\mathsf{U_b}$.

for each $\langle x, y \rangle \in \mathcal{I}_k \times \mathcal{I}_k$ **do**
  if can-follow(y, x) **then** I-I$_k(x, y) \leftarrow \surd$;
  for each $p \in \mathcal{P}_k$ **do**
    if can-occur-between(p, x, y) **then** add p to I-P-I$_k(x, y)$;

Owing to space constraints, only the I-I and I-P-I transition tables for level 2 are given here (see Figure 4). As described in §2.5, we can substitute the single kink-point atom $\mathsf{U_b}$ with a set of kink tokens: $\{\mathsf{U_<}, \mathsf{U_>}, \mathsf{U_\prec}, \mathsf{U_\succ}\}$. In order to do this, we need to replace each $\mathsf{U_b}$ in cell (x, y) of the I-P-I table with the correct subset of kink tokens, as follows: (i) replace $\mathsf{U_b}$ with $\mathsf{U_<}$ and $\mathsf{U_>}$, and (ii) add $\mathsf{U_\prec}$ if $\partial^2 x = +$ or $\partial^2 y = +$, and add $\mathsf{U_\succ}$ if $\partial^2 x = -$ or $\partial^2 y = -$, i.e., inward-pointing cusps must be flanked by an interval of positive curvature and outward-pointing cusps by an interval of negative curvature (cf. [Galton and Meathrel, 1999, §3.2]).

	P	Z	N
P		$\surd$	
Z	$\surd$		$\surd$
N		$\surd$	

	P	Z	N
P	Z $\mathsf{U_b}$ $\mathsf{U_c}$	$\mathsf{U_b}$ $\mathsf{U_c}$	Z $\mathsf{U_b}$ $\mathsf{U_c}$
Z	$\mathsf{U_b}$ $\mathsf{U_c}$	$\mathsf{U_b}$	$\mathsf{U_b}$ $\mathsf{U_c}$
N	Z $\mathsf{U_b}$ $\mathsf{U_c}$	$\mathsf{U_b}$ $\mathsf{U_c}$	Z $\mathsf{U_b}$ $\mathsf{U_c}$

Figure 4: The I-I and I-P-I tables for level 2

4.3 Graph correspondence

A TOG consists of nodes representing atomic tokens and edges representing token ordering constraints. The nodes for interval atoms are shown as rectangles and those for point atoms as circles. Any path through a TOG, which starts and finishes at a node representing an interval atom, yields a syntactically valid string of tokens that is instantiable as an open curve and that may or may not be instantiable as a closed curve. A TOG that encodes the ordering constraints for the atoms of level 2 (incorporating the set of kink tokens) is shown in Figure 5. We have seen that we can construct I-I and I-P-I tables for any level of the atomic hierarchy. From the tables for a particular level, a corresponding TOG can be constructed that visually encodes the constraints given by the table contents. We do not here provide an algorithm for constructing TOGs from I-I and I-P-I tables; details relevant to such an algorithm may be found in [Meathrel and Galton, 2000, §5]. The TOG in Figure 5, then, corresponds to the I-I and I-P-I tables given in Figure 4 (with $\mathsf{U_b}$ replaced with the kink tokens, as described at the end of §4.2). The TOGs in Figures 2 and 4 of [Meathrel and Galton, 2000] correspond, respectively, to the I-I and I-P-I tables of levels 2 and 3 of the atomic hierarchy[3].

[3] With $\mathsf{U_b}$ *not* replaced by the more-specific kink tokens.

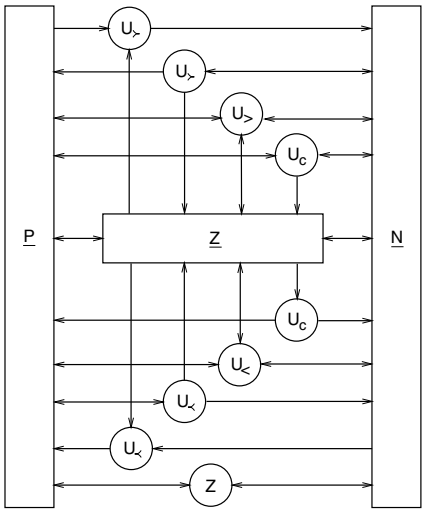


Figure 5: Level-2 TOG with distinct angle and cusp tokens

5 Conclusion

We have generalised the work described in [Meathrel and Galton, 2000], by deriving an unbounded hierarchy of atomic tokens and incorporating distinct kink tokens for inward and outward pointing angles and cusps. Both open and closed curves have qualitative descriptions, consisting of strings of atoms, at every level of the hierarchy, with the most coarse-grained description at level 1. Higher-numbered levels, where more qualitative components are included, possess greater discriminatory power and allow for finer-grained descriptions. We showed how, for each level of the hierarchy, I-I and I-P-I transition tables, encoding the ordering constraints for the set of atoms at that level, can be constructed by a simple algorithm that makes use of a pair of compatibility matrices. From a pair of I-I and I-P-I tables, a corresponding TOG can be constructed that visually encodes their constraints.

In [Meathrel and Galton, 2000], we showed how sequences of atomic tokens (with "contexts") can be used to model boundary-based schemes such as those of Leyton [1988] and Hoffman and Richards [1982]. We believe the theory presented here provides an adequate basis for constructing *any* boundary-based scheme of qualitative shape descriptors.

References

[Galton and Meathrel, 1999] A. Galton and R. Meathrel. Qualitative outline theory. In T. Dean, ed., *Proc. of 16th IJCAI*, pp. 1061–1066. Morgan Kaufmann, Inc., 1999.

[Hoffman and Richards, 1982] D. D. Hoffman and W. A. Richards. Representing smooth plane curves for recognition: implications for figure-ground reversal. In *Proc. of AAAI-82*, pp. 5–8, 1982.

[Leyton, 1988] Michael Leyton. A process-grammar for shape. *Artificial Intelligence*, 34:213–247, 1988.

[Meathrel and Galton, 2000] R. Meathrel and A. Galton. Qualitative representation of planar outlines. In W. Horn, ed., *Proc. of 14th ECAI*, pp. 224–228. IOS Press, 2000.

Resolving Ambiguities to Create a Natural Computer-Based Sketching Environment

Christine Alvarado, Randall Davis
MIT Artificial Intelligence Laboratory

Abstract

Current computer-based design tools for mechanical engineers are not tailored to the early stages of design. Most designs start as pencil and paper sketches, and are entered into CAD systems only when nearly complete. Our goal is to create a kind of "magic paper" capable of bridging the gap between these two stages. We want to create a computer-based sketching environment that feels as natural as sketching on paper, but unlike paper, understands a mechanical engineer's sketch as it is drawn. One important step toward realizing this goal is resolving ambiguities in the sketch— determining, for example, whether a circle is intended to indicate a wheel or a pin joint—and doing this as the user draws, so that it doesn't interfere with the design process. We present a method and an implemented program that does this for freehand sketches of simple 2-D mechanical devices.

1 Sketching Conceptual Designs

Engineers typically make several drawings in the course of a design, ranging from informal sketches to the formal manufacturing drawings created with drafting tools. Drawing is far more than an artifact of the design process; it has been shown to be essential at all stages of the design process [Ullman *et al.*, 1990]. Yet almost all early drawings are still done using pencil and paper. Only after a design is relatively stable do engineers take the time to use computer aided design or drafting tools, typically because existing tools are too difficult to use for the meager payoff they provide at this early stage.

Our aim is to allow designers to sketch just as they would on paper, e.g., without specifying in advance what component they are drawing, yet have the system understand what has been sketched. We want to have the input be as unconstrained as possible, in order to make interaction easy and natural; our route to accomplishing this is to build a sufficiently powerful sketch recognizer.

It is not yet obvious that a freehand sketching interface will be more effective in real use than a carefully designed menu-based system. In order to do the comparison experiments, however, we must first build powerful sketch-based systems. It is the construction of such a system that is the focus of this paper.

The value of sketching as an interface and the utility of intelligent sketch understanding has gained increasing attention in recent years (e.g., [Hearst, 1998]). Some early research was concerned with single stroke classification ([Rubine, 1991]), while more recent work ([Gross, 1995; Landay and Myers, 2001]) puts groups of strokes together to form larger components. A number of efforts (e.g., [Gross and Do, 1996], [Mankoff *et al.*, 2000]) have acknowledged the necessity of representing ambiguities that arise in interpreting strokes, but have not substantially addressed how to resolve those ambiguities.

Given the frequency of ambiguities in a sketch, a tool that constantly interrupts the designer to ask for a choice between multiple alternatives would be cumbersome. Our work is thus focused, in part, on creating a framework in which to both represent and use contextual (top-down) knowledge to resolve the ambiguities. We built a program called ASSIST (A Shrewd Sketch Interpretation and Simulation Tool) that interprets and understands a user's sketch as it is being drawn, providing a natural-feeling environment for mechanical engineering sketches.

The program has a number of interesting capabilities.

- The basic input to the program is a sketch, i.e., a sequence of strokes drawn "while the system watches," not a finished drawing to be interpreted only after it is complete.
- Sketch interpretation happens in real time, as the sketch is being created.
- The program allows the user to draw mechanical components just as on paper, i.e., as informal sketches, without having to pre-select icons or explicitly identify the components.
- The program uses a general architecture for both representing ambiguities and adding contextual knowledge to resolve the ambiguities.
- The program employs a variety of knowledge sources to resolve ambiguity, including knowledge of drawing style and of mechanical engineering design.
- The program understands the sketch, in the sense that it recognizes patterns of strokes as depicting particular

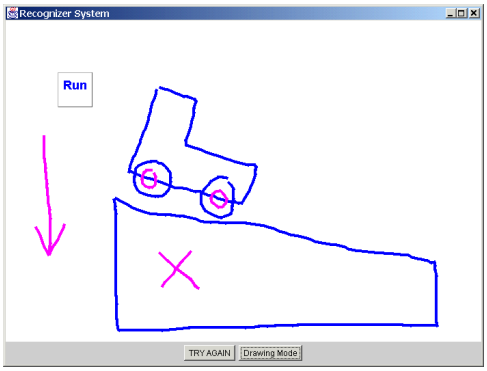


Figure 1: A car on a hill, as drawn by the user in ASSIST.

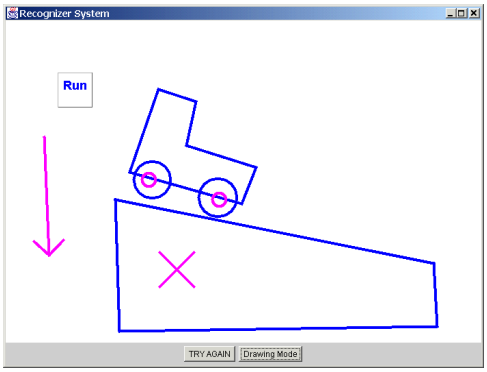


Figure 2: The sketch as displayed by ASSIST.

components, and illustrates its understanding by running a simulation of the device, giving designers a way to simulate their designs as they sketch them.

We describe the system and report on a pilot user study evaluating the naturalness of the program's interface and the effectiveness of its interpretations.

2 Designing with ASSIST

Figure 1 shows a session in which the user has drawn a simple car on a hill. The user might begin by drawing the body of the car, a free-form closed polygon. As the user completes the polygon, the system displays its interpretation by replacing the hand-drawn lines (shown in Figure 1) with straight blue lines. Next the user might add the wheels of the car, which also turn blue as they are recognized as circular bodies. The user can then "attach" the wheels with pin joints that connect wheels to the car body and allow them to rotate. The user might then draw a surface for the car to roll down, and anchor it to the background (the "x" indicates anchoring; anything not anchored can fall). Finally, the user can add gravity by drawing a downward pointing arrow not attached to any object. The user's drawing as re-displayed by ASSIST is shown in Figure 2.

The system recognizes the various components in the drawing by their form and context; when the "Run" button is tapped, it transfers the design to a two-dimensional mechanical simulator which shows what will happen (Figure 3).[1]

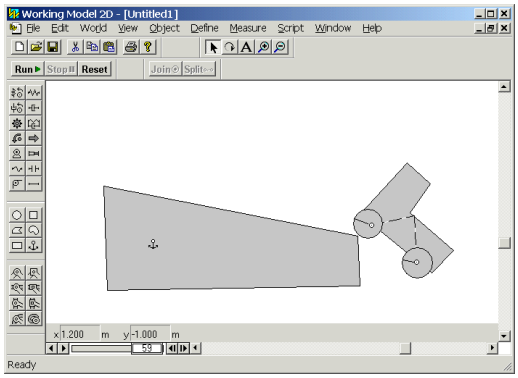


Figure 3: The sketch simulated, showing the consequences.

Note that the user drew the device without using icons, menu commands, or other means of pre-specifying the components being drawn. Note, too, that there are ambiguities in the sketch, e.g., both the wheels of the car and pin joints are drawn using circles, yet the system was able to select the correct interpretation despite these ambiguities, by using the knowledge and techniques discussed below. The automatic disambiguation allowed the user to sketch without interruption.

Figure 4 shows a session in which the user has drawn a more interesting device, a circuit breaker, and run a simulation of its behavior.

Note that ASSIST deals only with recognizing the mechanical components in the drawing and is, purposely, literal-minded in doing so. Components are assembled just as the user drew them, and component parameters (e.g. spring constants, magnitudes of forces, etc) are set to default values. The car in Figures 1–3, for example, wobbles as it runs down the hill because the axles were not drawn in the center of the wheels. The combination of literal-minded interpretation and default parameter values can produce device behavior other than what the user had in mind. Other work in our group has explored the interesting and difficult problem of communicating and understanding the *intended* behavior of a device once it has been drawn using ASSIST [Oltmans, 2000].

3 Embedding Intelligent Assistance

We created a model for sketch understanding and ambiguity resolution inspired by the behavior of an informed human observer, one that recognizes the sketch by relying on both low-level (i.e., purely geometric) routines and domain specific knowledge.

One interesting behavior of an informed observer is that interpretation begins as soon as the designer begins sketching. While not a required strategy—people can obviously interpret a finished sketch—there are advantages in ease of use and in speed from having the program do its interpretation in parallel with drawing. Ease of use arises because the program

[1] We use Working Model 2D from Knowledge Revolution, a commercial mechanical simulator; any simulator with similar capabilities would do as well.

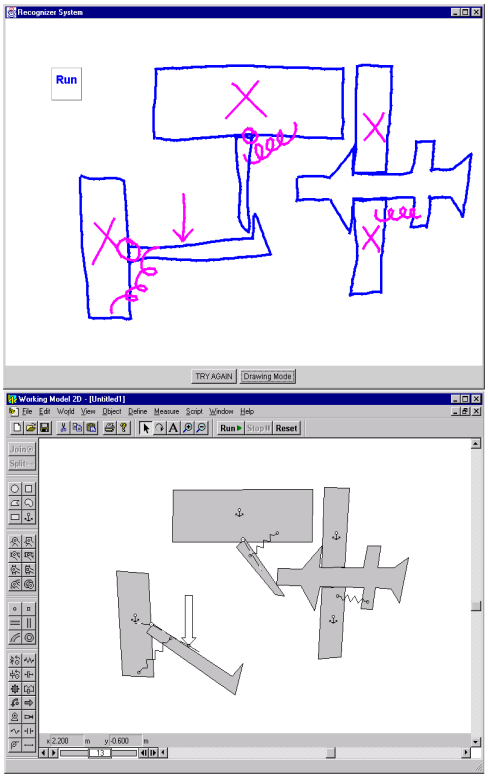


Figure 4: A sketch of a circuit breaker (left) and its simulation (right).

can provide an indication of its interpretation of parts of the sketch as soon as they are drawn, making it easier for the user to correct a misinterpretation. Interpretation is faster because incremental interpretation effects a divide and conquer strategy: parts of the drawing interpreted correctly can provide useful context when interpreting parts drawn subsequently.[2]

A second interesting behavior of an informed observer is the ability to accumulate multiple interpretations and defer commitment. Consider for example the objects in Figure 5. Are the strokes in 5a going to become part of a ball and socket mechanism (5b), or are they the beginning of a gear (5c)? Committing too soon to one interpretation precludes the other. Hence interpretation must be capable of revision in the face of new information.

There is clearly a need to balance out the desire for interpretation occurring in parallel with drawing, and the need to avoid premature commitment. We discuss below how our system accomplishes this.

Third, while commitment should be deferred, it must of course be made eventually, and determining when to make that commitment is not easy. Timing information can assist. Consider the case of circles: Because circles are low-level structures, it is likely that they will be used in higher-level structures, as for example when a circle turns out to be part of a pulley system. One way of dealing with this is to use timing

[2]The program also seems faster because it is working while the user is drawing, reducing the user's wait.

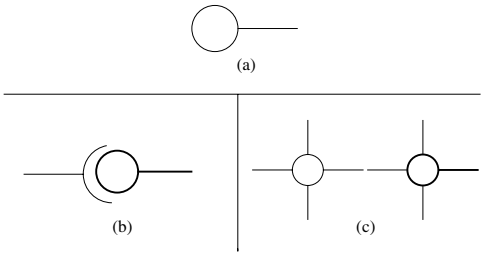


Figure 5: An example of ambiguity: The bold strokes in (b) and (c) are identical to the strokes in (a).

data: the system gets to "watch" the sketch being drawn and knows when each stroke was made. If, some time after the circle has been drawn, it has still not been used in any other structure, the observer can plausibly guess that it will not be incorporated into another piece and should be interpreted as an independent circular body.[3]

Finally, parts may remain ambiguous even when a piece of the drawing is finished. To resolve these residual ambiguities, the observer uses his knowledge of mechanical engineering components and how they combine. Consider, for example, the small circles inside the larger circles in Figure 2; ASSIST determines that these are more likely to be pivot joints than additional circular bodies, both because small circles typically indicate pin joints and because bodies do not typically overlap without some means of interconnection (i.e., the pin joint).

Our system incorporates each of these observations: it begins interpreting the sketch as soon as the user starts drawing; it accumulates multiple interpretations, deferring commitment until sufficient evidence (e.g., stroke timing) accumulates to suggest a component has been finished, and it resolves ambiguities by relying on knowledge from the domain about how components combine.

4 ASSIST's Interpretation and Disambiguation Process

ASSIST's overall control structure is a hierarchical template-matching process, implemented in a way that produces continual, incremental interpretation and re-evaluation as each new stroke is added to the sketch. Each new stroke triggers a three stage process of recognition, reasoning and resolution. Recognition generates all possible interpretations of the sketch in its current state, reasoning scores each interpretation, and resolution selects the current best consistent interpretation. After each pass through the three stages the system displays its current best interpretation by redrawing the sketch.

4.1 Recognition

In the recognition stage, ASSIST uses a body of recognizers, small routines that parse the sketch, accumulating all possible

[3]A *body* is any hunk of material not otherwise interpreted as a more specialized component (like a spring, pin joint, etc.). The car body is a polygonal body; its wheels are circular bodies.

interpretations as the user draws each stroke. A recognizer takes as input raw strokes and previously recognized objects, and if the input fits its template, produces a new object. For example, the circle recognizer reports a circle if all the points on a stroke lie at roughly the same distance from the average X and Y coordinate of the stroke.[4] The circle is then available to other recognizers, e.g., the pulley recognizer.

4.2 Reasoning

In the second stage the system scores each interpretation using a variety of different sources of knowledge that embody heuristics about how people draw and how mechanical parts combine.

Temporal Evidence

People tend to draw all of one object before moving to a new one. Our system considers interpretations that were drawn with consecutive strokes to be more likely than those drawn with non-consecutive strokes.

Additional evidence comes from "longevity:" the longer a figure stays unchanged, the stronger its interpretation becomes, because as time passes it becomes more likely that the figure is not going to be turned into anything else by additional strokes.

Simpler Is Better

We apply Occam's razor and prefer to fit the fewest parts possible to a given set of strokes. For example, any polygonal body (e.g., the car body in Figure 2) could have been interpreted as a set of (connected) individual rods, but the system prefers the interpretation "body" because it fits many strokes into a single interpretation.

More Specific is Better

Our system favors the most specific interpretation. Circles, for example, (currently) have three interpretations: circular bodies, pin joints, and the "select" editing gesture. The selection gesture is the most specific interpretation, in the sense that every circle can be a circular body or pin joint, but not every circle can be a selection gesture (e.g., if it does not encircle any objects). Hence when a circle contains objects inside of it, the system prefers to interpret it as a selection gesture.

Domain Knowledge

ASSIST uses basic knowledge about how mechanical components combine. For example, a small circle drawn on top of a body is more likely to be a pin joint than a circular body.

User Feedback

User feedback also supplies guidance. The "Try Again" button (see the bottom of Figure 1) permits the user to indicate that something was recognized incorrectly, at which point the system discards that interpretation and offers the user an ordered list of alternative interpretations. Conversely the system can be relatively sure an interpretation is correct if the user implicitly accepts it by continuing to draw.

[4]In other work we describe recognizers that use more sophisticated early processing of basic geometry [Sezgin,].

Combining Evidence

The heuristics described above all independently provide evidence concerning which interpretation is likely to be correct. Our method of combining these independent sources involves distinguishing between two categories of evidence: categorical and situational.

Categorical evidence ranks interpretations relative to one another based on the first four knowledge sources described above. Each source is implemented in the system as a set of rules that takes two interpretations as input, and outputs an ordering between them. In processing Figure 1, for example, the interpretation "body" is ranked higher than the interpretation "connected rods," based on the "Simpler is Better" heuristic.

Situational evidence comes from implicit and explicit feedback from the user. Explicit feedback is provided by use of the "Try Again" button; implicit feedback arises when the user keeps drawing after the system displays an interpretation, suggesting that the user is satisfied that the system has understood what has been drawn so far.

The system gives each interpretation two numeric scores, one from each category of evidence. The categorical score is an integer from 0 to 10; the situational score is an integer from -11 to 11. These values are chosen so that the situational dominates the categorical, because we want user feedback to dominate general ranking rules. An interpretation's total score is simply the sum of its two scores.

To convert categorical evidence to a numerical score (so it can be combined it with the situational score), we generate a total ordering of all the interpretations consistent with the partial orders imposed by the categorical evidence. We do a topological sort of the graph of partial orders produced by the evidence and distribute scores evenly, from 0 to 10, over all the interpretations in the sorted graph.[5]

Situational scores start out at 0 and are strengthened or weakened by evidence that can raise of lower the current value by 1 or by 11. Situational evidence thus either modifies an interpretation's value by a small amount (1 unit) or declares it to be certainly correct or certainly incorrect. The system declares an interpretation to be certainly correct or certainly incorrect when the user explicitly accepts or rejects the interpretation using the "Try Again" dialog box. The system strengthens an interpretation by a small amount each time strokes added by the user are consistent with that interpretation.[6]

We developed this approach to accumulating and combining evidence, and implemented our knowledge sources as a rule based system, in order to provide a degree of modularity

[5]The system first removes cycles in the graph by collapsing strongly connected components. Conceptually, this step indicates that the system will give an equal score to all interpretations that have inconsistent ordering given the evidence (i.e., one rule says A is more likely than B, while another says B is more likely than A). In addition, if there are more than 11 interpretations, the top ten are assigned scores of 10 through 1; the remaining interpretations all receive a score of 0.

[6]The system does not yet weaken an interpretation by a small amount; we have included this possibility for symmetry and possible future use.

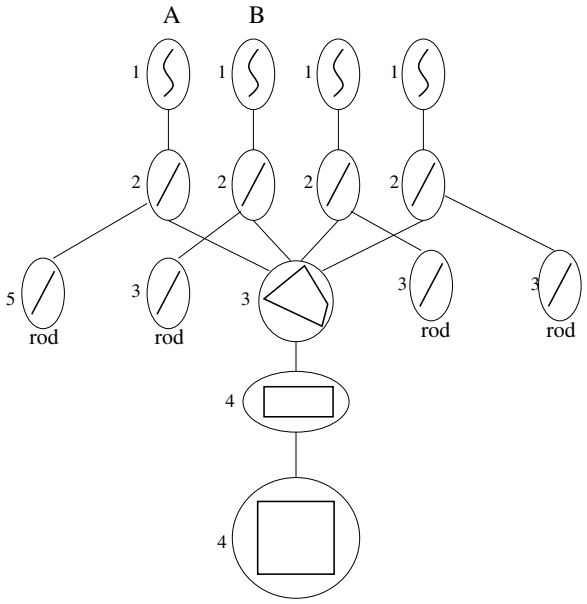


Figure 6: A recognition graph for four strokes; scores are shown at the left of each interpretation.

to the system. Our overall approach to the problem is to take into account as many sources of knowledge as prove useful in interpreting the sketch. We knew that it would be impossible to identify and implement them all at the outset, hence our design put a high premium on the ability to add and remove sources of evidence easily.

4.3 Resolution

The third stage in the interpretation process involves deciding which interpretation is currently the most likely. Our system uses a greedy algorithm, choosing the interpretation with the highest total score, eliminating all interpretations inconsistent with that choice, and repeating these two steps until no more interpretations remain to be selected.

The process is illustrated by the interpretation graph in Figure 6, which shows in graphical form all of the possible interpretations of four strokes (the top row of ovals): 4 separate lines, 4 rods, a quadrilateral, rectangle, or square. The rod on the left has the highest score, so it is chosen as a correct interpretation for stroke A. Choosing that interpretation eliminates the interpretations of quadrilateral, rectangle or square, because stroke A is needed in any of these interpretations. In this context the other strokes are interpreted as rods because that interpretation has the highest score of any remaining interpretation.

Recall that our interpretation process is continuous: all three stages of processing occur after every new stroke is added to the sketch, and the current best interpretation as selected by the greedy algorithm is presented to the user. The process tends to settle down reasonably quickly, in part because, as noted, we reward longevity. Hence once an interpretation has been presented to the user and unchanged for some period of time, it becomes increasingly unlikely to change.

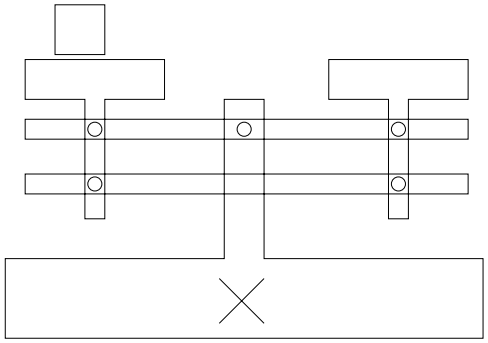


Figure 7: A scale.

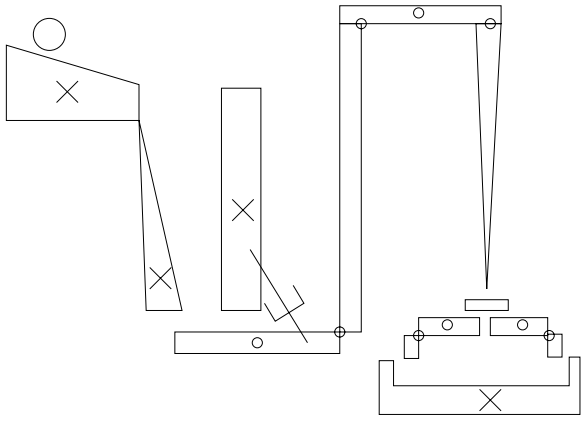


Figure 8: A Rube-Goldberg machine. The ball rolling down the incline sets in motion a sequence of events that eventually pushes the block at the right into the receptacle at bottom right. The device is an adaptation of the one found in [Narayanan, 1995].

5 Evaluation and Results

Our initial evaluation of ASSIST has focused on its naturalness and effectiveness. We asked subjects to sketch both on paper and using ASSIST. We observed their behavior and asked them to describe how ASSIST felt natural and what was awkward about using it.

We tested the system on eleven people from our the laboratory, two of whom had mechanical engineering design experience. All were asked first to draw a number of devices on paper (Figures 7, 8, 9), to give them a point of comparison and to allow use to observe differences in using the two media.

They were then asked to draw the same systems using ASSIST (they drew with a Wacom PL-400 tablet, an active matrix LCD display that allows users to sketch and see their strokes appear directly under the stylus). We asked them how often they felt the system got the correct interpretation and how reasonable the misinterpretations were, and asked them to compare using our system to drawing on paper and to using a menu-based interface.

The system was successful at interpreting the drawings despite substantial degrees of ambiguity, largely eliminating the

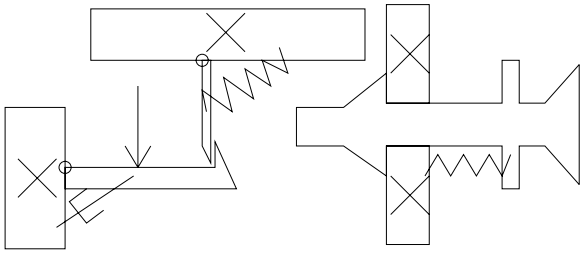


Figure 9: A circuit breaker.

need for the user to specify what he was drawing. As a consequence, a user's drawing style appeared to be only mildly more constrained than when drawing on paper.

People reported that the system usually got the correct interpretation of their sketch. Where the system did err, examination of its performance indicated that in many cases the correct interpretation had never been generated at the recognition step, suggesting that our reasoning heuristics are sound, but we must improve the low-level recognizers. This work is currently under way.

Users tended to draw more slowly and more precisely with ASSIST than they did on paper. The most common complaint was that it was difficult to do an accurate drawing because the system changed the input strokes slightly when it re-drew them (to indicate its interpretations). Users felt that the feedback given by ASSIST was effective but at times intrusive. Our next generation of the system leaves the path of the strokes unchanged, changing only their color to indicate the interpretation.

For a more complete discussion responses to the system from a user interface perspective, see [Alvarado and Davis, 2001].

6 Related Work

The Electronic Cocktail Napkin (ECN) project [Do and Gross, 1996; Gross, 1996] attacks a similar problem of sketch understanding and has a method for representing ambiguity. Our system takes a more aggressive approach to ambiguity resolution and as a result can interpret more complicated interactions between parts. In order for ECN to to resolve ambiguity, the user must either inform the system explicitly of the correct interpretation, or the system must find a specific higher-level pattern that would provide the context to disambiguate the interpretation of the stroke. Our system, in contrast, takes into account both drawing patterns and knowledge of drawing style.

[Mankoff *et al.*, 2000] presents a general framework for representing ambiguity in recognition-based interfaces. This work is similar in using a tree-like structure for representing ambiguity, but touches only briefly on ambiguity resolution. Our work pushes these ideas one step further within the domain of mechanical engineering by providing a framework and set of heuristics for ambiguity resolution.

SILK [Landay and Myers, 2001] allows a user to sketch out rough graphical user interface designs, then transform them into more polished versions. SILK addresses the notion of ambiguity, but limits its handling of it to single parts, e.g., is this group of strokes a radio button or a check box? This does not in general affect the interpretation of the other strokes in the sketch. In contrast, our system can resolve ambiguities that affect the interpretation of many pieces of the sketch.

A theoretical motivation to our work was provided by work in [Saund and Moran, 1995], which outlines several goals in interpreting ambiguous sketches. Our work implements many of the multiple representation and disambiguation techniques suggested in their work.

We have also been motivated by work in mechanical system behavior analysis, especially in the field of qualitative behavior extraction and representation [Sacks, 1993; Stahovich *et al.*, 1998]. The work by Stahovich aims to extract the important design constraints from the designer's rough sketch and is less focused on the interface or sketch recognition process. It was nevertheless the inspiration for our work in this area.

7 Future Work

The work presented in this paper is a first step toward creating a natural interface. It can usefully be expanded in several areas.

First, our current formulation of recognition and evidential reasoning is of course quite informal. This is a consequence of our focus at this stage on the knowledge level, i.e., trying to determine what the program should know and use to evaluate interpretations. Once the content has become more stable and better understood, a more formal process of evaluation and control (e.g., Bayes' nets) may prove useful both for speed and scaling.

Second, in our efforts to combine the best properties of paper and the digital medium we have yet to find many of the appropriate trade-off points. How aggressive should the system be in its interpretations? Forcing the user to correct the system immediately when it makes a mistake greatly aids recognition, but may distract the designer by forcing her to focus on the system's recognition process rather than on the design. In addition, some ambiguities are resolved as more of the sketch is drawn, yet if the system waits for the sketch to be finished, unraveling an incorrect interpretations can be a great deal of work.

In the same vein, it will be important to calibrate how important true freehand sketching is to designers. The obvious alternative is a icon-based system with graphical editing capabilities (e.g., moving and resizing the standard components). Freehand drawing can be powerful, but alternative interface styles need to be considered as well.

The system should also adapt to new users and their sketching style. For example, one of our heuristics was that people draw all of one object before moving onto the next, but there are of course exceptions. The system should be able to adjust to this type of behavior and learn to override its default heuristic.

8 Conclusion

CAD systems are rarely used in early design because they do not allow for quick and natural sketching of ideas. To be

useful here, computers must allow the designer to sketch as on paper, yet provide benefits not available with paper, such as the ability to simulate the system.

To provide an interface that feels natural yet interprets sketches as the user draws, the system must be able to resolve ambiguities without interrupting the user. This work provides one solution to problem of ambiguity resolution in a framework of reasonable generality.

Acknowledgments

Luke Weisman and Mike Oltmans helped extensively in the development of these ideas and this system. The work was supported in part by the National Science Foundation, the MIT/Ford Motor Company Collaboration, and MIT's Project Oxygen.

References

[Alvarado and Davis, 2001] Christine Alvarado and Randall Davis. Preserving the freedom of sketching to create a natural computer-based sketch tool. In *Human Computer Interaction International Proceedings*, 2001.

[Do and Gross, 1996] Ellen Yi-Luen Do and Mark D. Gross. Drawing as a means to design reasoning. *AI and Design*, 1996.

[Gross and Do, 1996] Mark Gross and Ellen Yi-Luen Do. Ambiguous intentions: a paper-like interface for creative design. In *Proceedings of UIST 96*, pages 183–192, 1996.

[Gross, 1995] Mark D. Gross. Recognizing and interpreting diagrams in design. In *2nd Annual International Conference on Image Processing*, pages 308–311, 1995.

[Gross, 1996] Mark D. Gross. The electronic cocktail napkin - a computational environment for working with design diagrams. *Design Studies*, 17:53–69, 1996.

[Hearst, 1998] Marti Hearst. Sketching intelligent systems. *IEEE Intelligent Systems*, pages 10–18, May/June 1998.

[Landay and Myers, 2001] James A. Landay and Brad A. Myers. Sketching interfaces: Toward more human interface design. *IEEE Computer*, 34(3):56–64, March 2001.

[Mankoff *et al.*, 2000] Jennifer Mankoff, Scott E Hudson, and Grefory D. Abowd. Providing intergrated toolkit-level support for ambiguity in recogntiion-based interfaces. In *Proceedings of the CHI 2000 conference on Human factors in computing systems*, pages 368–375, 2000.

[Narayanan *et al.*, 1995] N. Hari Narayanan, Masaki Suwa, and Hiroshi Motoda. *Behavior Hypothesis from Schematic Diagrams*, chapter 15, pages 501–534. The MIT Press, Cambridge, Massachusetts, 1995.

[Oltmans, 2000] Michael Oltmans. Understanding natually conveyed explanations of device behavior. Master's thesis, Massachusetts Institute of Technology, 2000.

[Rubine, 1991] Dean Rubine. Specifying gestures by example. *Computer Graphics*, pages 329–337, July 1991.

[Sacks, 1993] Elisha Sacks. Automated modeling and kinematic simulation of mechanisms. *Computer-Aided Design*, 25(2):107–118, 1993.

[Saund and Moran, 1995] Eric Saund and Thomas P. Moran. Perceptual organization in an interactive sketch editing application. In *ICCV 1995*, 1995.

[Sezgin,] Metin Sezgin. Early processing in sketch understanding. Unpublished Master's Thesis, Massachusetts Institute of Technology.

[Stahovich *et al.*, 1998] T. Stahovich, R. Davis, and H.Shrobe. Generalting multiple new designs from a sketch. *Artificial Intelligence*, 104(1-2):211–264, 1998.

[Ullman *et al.*, 1990] David G. Ullman, Stephan Wood, and David Craig. The importance of drawing in mechanical design process. *Computer & Graphics*, 14(2):263–274, 1990.

ROBOTICS AND PERCEPTION

VISION I

VAMBAM: View and Motion -based Aspect Models for Distributed Omnidirectional Vision Systems

Hiroshi Ishiguro* and Takuichi Nishimura**
*Department of Computer & Communication Sciences, Wakayama University, Japan
** Cyber Assist Research Center,
National Institute for Advanced Industrial Science and Technology (AIST)
*ishiguro@sys.wakayama-u.ac.jp, ** t_nishimura@aist.go.jp

Abstract

This paper proposes a new model for gesture recognition. The model, called view and motion -based aspect models (VAMBAM), is an omni-directional view-based aspect model based on motion-based segmentation. This model realizes location-free and rotation-free gesture recognition with a distributed omnidirectional vision system (DOVS). The distributed vision system consisting of multiple omnidirectional cameras is a prototype of a perceptual information infrastructure for monitoring and recognizing the real world. In addition to the concept of VABAM, this paper shows how the model realizes robust and real-time visual recognition of the DOVS.

1 Introduction

An open problem of visual recognition is “segmentation” of sensory data. Based on Marr’s paradigm [Marr, 1982], many researchers have tried to build recognition systems using 3-D models as the intermediate representations. There are mainly two methods. One is to detect visual features such as line segments and surfaces bounded by the lines and then to directly fit wire-frame models to the detected features or by estimating 3-D positions of the features by using multiple views. Another more powerful method is to detect targets by multiple-camera stereo [Kanade, xxxx]. The target segmentation by cameras surrounding the target is robust against lighting conditions. A problem of the former method is in the feature segmentation in the image. It has a serious problem for deformable targets, such as humans. A problem of the later method is in the computational const. Even if we use the most powerful computer, it is difficult to compute in real-time.

Fig. 1: Omnidirectional camera

The purpose of this paper is to develop a real-time vision system that tracks and recognizes human behaviors

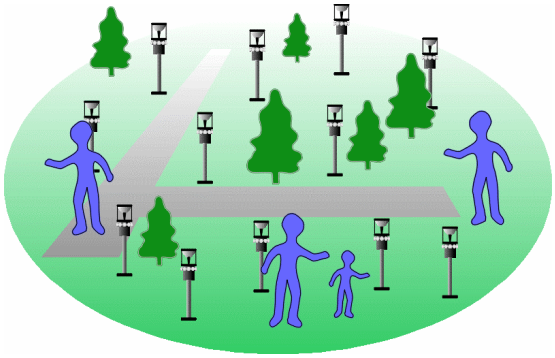


Fig. 2: Distributed omnidirectional vision system

in a wide area. Therefore, the methods based on the 3-D models are not suitable for our purpose. We, rather, take a new but well-known memory-based approach for the segmentation problem. Recent progress of computer hardware enabled us to use a large volume of memory and to access in real time. This paper follows the basic idea and proposes a general framework of visual recognition in a wide environment.

The system introduced in this paper consists of omnidirectional cameras. Each camera has an omnidirectional visual field and observes a wide range. Fig. 1 shows a compact omnidirectional camera [Ishiguro, 1998]. By using multiple omnidirectional cameras, the system redundantly covers the wide environment as shown in Fig. 2. We call it *Distributed Omnidirectional Vision System* (DOVS). The goal of our research is to develop a *perceptual information infrastructure* [Ishiguro, 1997] by using the distributed omnidirectional vision system as an extension of computer and sensor networking.

The key function of the DOVS is to track walking humans and to recognize the gestures in real time. That is, it is *location-free and rotation-free gesture recognition*. This paper proposes a new model of visual recognition for realizing the function. We call the model *View And Motion -Based Aspect Model* (VAMBAM). The model has two features as follows:

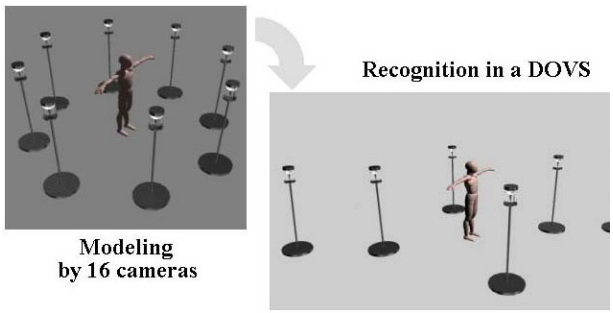


Fig. 3: Modeling and recognition

1. Omnidirectional model of a gesture
2. Motion-based segmentation of the gesture

As shown in Fig. 3, the multiple cameras observe a human from various viewing point to build the VAMBAM. Then, the system recognizes human gestures in the distributed vision system that consists of multiple cameras distributed in the environment. A VAMBAM maintains the visual information in a form of view-based aspect model. In the modeling, a gesture is observed as image sequences. Then, the image sequence is transformed to a feature vector that represents an aspect of the gesture based on the motion information. Thus, VAMBAM maintains omnidirectional and motion-based gesture information. This gesture recognition not using 3-D models has not been developed so far [Pavlovic, 1997].

In the following sections, Section 2 explains the design policies of VAMBAM. To utilize VAMBAM, we have developed a 3-D dynamic programming matching. Section 3 describes the detail. Finally, Section 4 reports experimental results and effectiveness of VAMBAM.

2 VAMBAM

2.1 Omnidirectional Aspect Model

The system has two phases for recognizing human gestures: modeling phase and recognition phase (see Fig. 3). In the modeling phase, the system acquires VAMBAMs by using 16 cameras surrounding a human. The human makes several gestures and the system memorizes each gesture as an individual VAMBAM. In the recognition phase, the system consisting of multiple omnidirectional cameras finds a human, tracks, and then matches acquired images from several cameras with the stored VAMBAMs. This section explains the modeling phase.

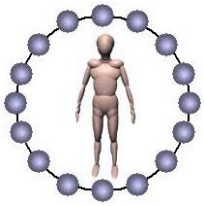


Fig. 4: 16 feature vectors

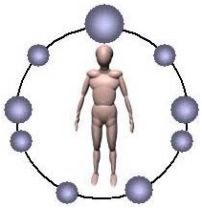


Fig. 5: Weighted 16 feature vectors

An ideal model for object recognition is to have all visual information from all directions like a complete 3-D model. If there are an infinite number of cameras surrounding an object, we can directly acquire such an ideal view-based model. VAMBAM is an approximation of such an ideal model.

A problem is how to reduce the number of cameras surrounding a modeling target in the modeling phase. It depends on complexity of the target. Since the purpose of the DOVS is to recognize coarse human gestures, we have decided the number as 16 based on a preparatory experiment.

By using 16 cameras, the system acquires 16 view sequences and transforms them to 16 feature vectors. In Fig. 4, the sphere shows a feature vector. Here, we suppose all of the cameras are located at the same height. Of course, we can extend the camera arrangement, but it is not needed for our current gesture recognition.

All 16 cameras do not have the same priority. For recognizing a particular gesture, for example breathing gesture, the front camera is more important than the side cameras. Therefore, each feature vector has a weight estimated from the motion information. If the camera observes more moving pixels by background subtraction than the others, it has a bigger weight. Fig. 5 shows the conceptual figure of the weight vectors. *The weights define aspect changes* in VAMBAM. Fig. 5 shows 9 feature vectors that have positive weights, and it represents 9 aspects. The system efficiently refers to the weight for searching the target in the recognition phase.

The VAMBAM shown in Fig. 5 is obtained by observing the target with a constant distance. If the camera is distant from the object, the aspect also changes. Bobick and Bolles [Bobick, 1992] reported an important idea that the model structure changes according to the resolution of a vision sensor. We follow the idea and

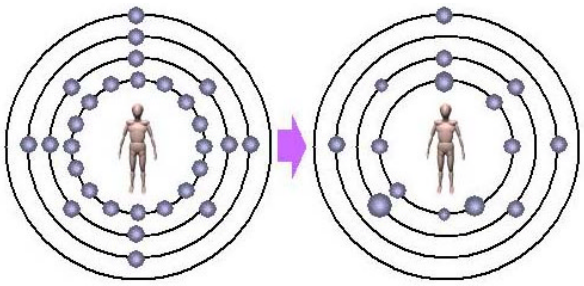


Fig. 6: Full VAMBAM

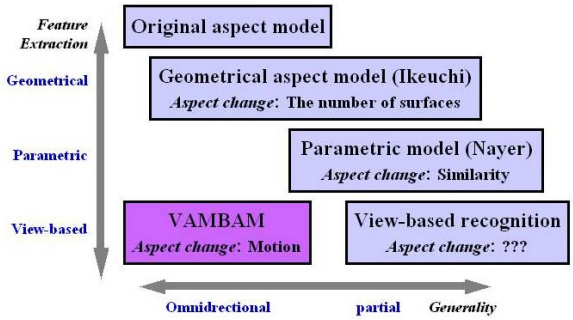


Fig. 7: Several approaches for visual modeling

extend a VAMBAM to a full VAMBAM as shown in Fig. 6. The full VAMBAM has several layers to handle different resolutions. In Fig. 6, the lateral circle represents the coarsest resolution (or distant observation). Although we have not developed the system using the Full VAMBAM, this concept is very important for our future extension of the basic model.

Here, let us state VAMBAM as an aspect model. Aspect models represent omnidirectional visual information surrounding the target. Koenderink [Koenderink, 1979] proposed the original concept based on 3-D structure of the object. Ikeuchi [Ikeuchi, 1988] modified for passive vision systems and defined the aspect change by the number of observable surfaces of the object. In their works, the important point is that the aspect model maintains omnidirectional visual information for the generality. On the other hand, view/appearance-based methods and parametric methods are also a kind of aspect model. The view-based method defines the aspect change as a difference among views by template matching or feature-based comparison. The parametric method defines the aspect change as a distance in the principle component parameter space. They, however, do not maintain the omnidirectional visual information in the previous works. VAMBAM is a view-based approach as described in the next subsection, but it is also an aspect model that maintains omnidirectional visual information. Fig. 7 shows relationships among the existing methods and our method.

2.2 Motion-based Segmentation

Another important feature of VAMBAM is motion-based segmentation of visual information. As mentioned before, the motion-based segmentation reduces the amount of memory and enables us to utilize omnidirectional view-based aspect models for practical systems.

In Fig. 4, each feature vector is given as shown in Fig. 8. The camera finds a moving object by background subtraction and tracks it as a candidate of a human. Then, it detects the center of gravity by making histograms along vertical and horizontal axes. The histograms give the height and width of the human. Based on the parameters, the system defines 9 regions that cover the

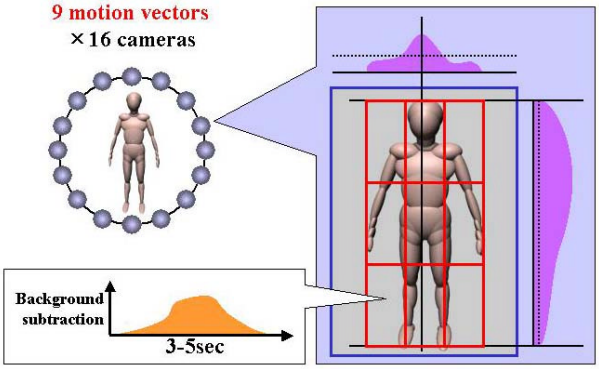


Fig. 8: Acquisition of a feature vector

human as shown in the left of Fig. 8, tracks the human fitting the regions. In each region, the system recodes the moving pixels detected by background subtraction and acquires 9 motion vectors in 3-5 sec. observation. Here, the observation length is adjusted according to the gesture. The acquired 9 motion vectors define a feature vector as represented with a sphere in Fig. 8. As mentioned before, several feature vectors do not contains motion information because of the viewing direction. By referring to the average and variance of the 9 motion vectors, the system assigns a weight to the feature vector.

The motion-based segmentation by background subtraction is one of the most robust and stable feature extractions and many practical systems use background subtraction. If we consider that "motion" represents relationships between static objects and events and gives meaning to the static objects, it is natural to build a recognition system based on the motion-based feature extraction.

Further, this representation of a gesture is compact. Suppose to record a 5 sec. gesture. A motion vector consists of 5×30 integers and 16 feature vectors have $5\times30\times9\times16=21600$ integers and the quarter is selectively memorized by the weights. If we represent an integer with 8bit, the necessary memory size will be 4000 bit and it is not large for recognizing a dozen of gestures in real time.

3 Two Techniques to Support VAMBAM

3.1 Real-time Tracking of Walking Humans

For realizing the location-free and direction-free gesture recognition, the system needs two more functions: one is tracking of walking humans in real time, the other is a robust matching method with gesture models.

By utilizing redundant visual information of the DOVS, we can build a robust and real-time multiple camera stereo system. This stereo, called *N-ocular stereo*, is an extension of trinocular stereo [Kitamura, 1990]. The basic process of N-ocular stereo is as follows:

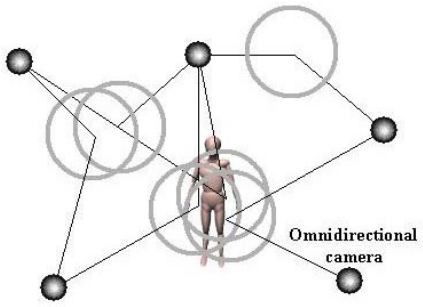


Fig. 9: N-ocular stereo by a DOVS

1. Detect moving regions on the images by background subtraction.
2. For all combination of two moving regions on different omnidirectional cameras, estimate the distance to the region and apply a circle as a human model (the circles in Fig. 9).
3. Check the overlapping of the circles and determine human locations.

As described above, we implement the N-ocular stereo as a model-based stereo method. Generally, the model-based stereo is more stable than the feature-based stereo in the case where the target is known. However, appearance of a human body frequently deforms on the image. Therefore, we have approximated the human body with a circle (the diameter is 60 cm) on the horizontal plane that is parallel to the floor. When three or more circles overlap each other, the system decides a human exists in the center of gravity of the circles. [Sogo, 2000, Sogo, 2001].

This N-ocular stereo for tracking humans in a DOVS is executed in real-time with a standard computer and it robustly tracks up to 3 humans simultaneously with four omnidirectional cameras. The reason of the robustness is in the redundant observation by multiple omnidirectional vision sensors. The method based on background subtraction is, generally speaking, noisy against change of lighting condition. However, the multiple observations from different viewing angles suppress the noise.

Another issue in the N-ocular stereo is precise localization and identification of the omnidirectional cameras. For this issue, we have already developed a method that automatically performs the localization and identification [Kato, 1999].

3.2 3-D Dynamic Programming for Matching

The recognition system needs to handle multiple gesture models simultaneously. Therefore, a robust and real-time matching technique is required.

The basic idea is to use conventional continuous dynamic programming (CDP). We have extended this CDP to interpolate discrete observation by 16 cameras, which is called *Spatio-Temporal Continuous Dynamic Programming* (STCDP, see Fig. 11). In the recognition phase, a stored model does not exactly match to the observations by the DOVS as shown in Fig. 10. The STCDP searches the best match by swiveling the model within 360/16 degrees. In addition to this, the STCDP finds the direction of target by rotating the models 360 degrees. Thus, this 3-D matching realizes the rotation-free gesture recognition by using the discrete model.

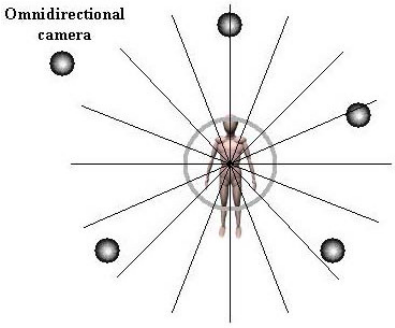


Fig. 10: Matching with a VAMBAM

By taking account of all conditions in the matching, we formalize the STCDP as follows. Let us denote the cumulative distance of a point (i_q, t, t) for evaluation of the DP matching as $S(i_q, t, t)$. Suppose the q th feature vector is rotated i_q up to N_q and the model length is T. The STCDP calculates $S(i_q, t, t)$ by using the following recursive equations.

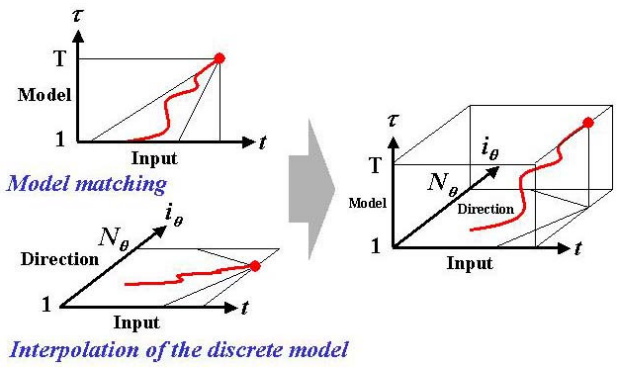


Fig. 11: STCDP

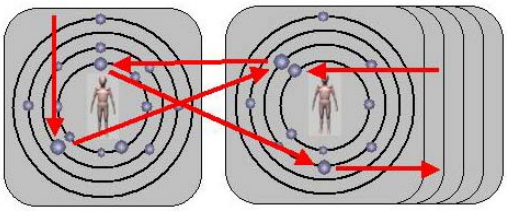


Fig. 12: Attention control in the matching

Boundary conditions $(1 \le i_q \le N_q, 1 \le t \le T, 0 \le t)$:

$$S(i_q, 0, t) = \infty$$
$$S(0, t, t) = S(N_q + 1, t, t) = \infty$$
$$S(i_q, t, -1) = S(i_q, t, 0) = \infty$$

Recursive equations $(1 \le t)$:

$$S(i_q, 1, t) = 3 \cdot d(i_q, 1, t)$$
$$S(i_q, t, t) =$$
$$\min \begin{cases} S(i_q, t-1, t-2) + 2 \cdot d(i_q, t, t-1) + d(i_q, t, t) \\ S(i_q, t-1, t-1) + 3 \cdot + d(i_q, t, t) \\ S(i_q, t-2, t-1) + 3 \cdot d(i_q, t-1, t) + 3 \cdot d(i_q, t, t) \\ S(i_q - 1, t-1, t-2) + 2 \cdot d(i_q - 1, t, t-1) + d(i_q, t, t) \\ S(i_q - 1, t-1, t-1) + 3 \cdot + d(i_q, t, t) \\ S(i_q - 1, t-2, t-1) + 3 \cdot d(i_q - 1, t-1, t) + 3 \cdot d(i_q, t, t) \\ S(i_q + 1, t-1, t-2) + 2 \cdot d(i_q + 1, t, t-1) + d(i_q, t, t) \\ S(i_q + 1, t-1, t-1) + 3 \cdot + d(i_q, t, t) \\ S(i_q + 1, t-2, t-1) + 3 \cdot d(i_q + 1, t-1, t) + 3 \cdot d(i_q, t, t) \end{cases}$$
$$(2 \le t \le T)$$

In the current implementation, the STCDP sequentially matches the input with 10 stored models. However, the STCDP needs to be modified to handle a large number of models. The weights assigned to the VAMBAM reduce the computational time drastically. By referring to the weights, the STCDP can dynamically select possible feature vectors and ignore redundant search for impossible models in the early stage as shown in Fig. 12. In other words, we can see attention control in the recognition phase using VAMBAM.

3 Experimental Results

For the modeling, we have arranged 16 cameras along a circle of 3m diameter and acquired ten model gestures: (a) deep-breathing, (b) turning around the upper body, (c) kicking, (d) throwing, (e) jumping, (f) opening two legs, (g) picking up, (h) sitting down, (i) bowing, (j) Sumo-stomping. Fig. 13 shows the modeling system.

Then, we have verified the performance of the proposed method by using 4 omnidirectional cameras. Fig. 14 shows the top view of 4 cameras tracking a walking human and Fig. 15 shows a part of the trajectory. Fig. 16 shows an image sequence while the system recognizes "kick" gesture. Fig. 17 shows the monitor output of the system.

As shown in Table 1, we could verify good performance of the system. In the recognition phase, we have prepared different humans with the modeling phase and they have randomly iterated the 10 gestures 10 times.

Fig. 13: Modeling system

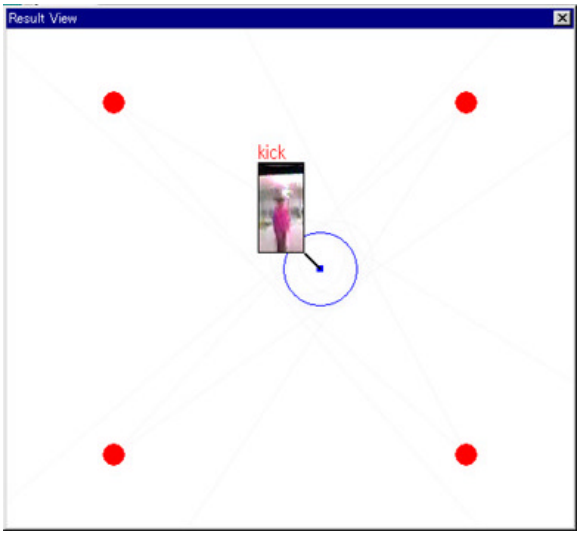


Fig. 14: Tracking of a walking human

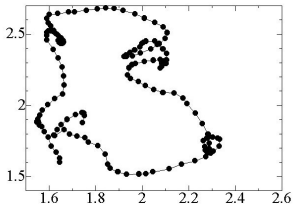


Fig. 15: Trajectory of the walking human

When the human behaves a gesture at an arbitrary location with an arbitrary orientation, the recognition results using STCDP was 100% with 4 cameras as shown in Table 1(a). Further, a human can behave similar behavior even if he/she is moving. In the case, the performance was 87%. In this experimentation, the system has recognized in real time.

The purpose of this system is to realize gesture recognition in a wide area. Although the experimental results are not sufficient, we could verify the basic function.

(a) Recognition results when the human stops

	CDP	STCDP
1 camera	65%	83%
4 cameras	100%	100%

(b) Recognition results when the human moves

	CDP	TSCDP
1 camera	31%	62%
4 cameras	64%	87%

Table 1: Recognition results

4 Conclusions

This paper has proposed a new model for gesture recognition, called VAMBAM. By using the model for the distributed omnidirectional vision system, we have developed a location-free and rotation-free gesture recognition system.

There are several remained problems yet. One is resolution of the gesture recognition. The current system handles only large gestures using arms and legs. However, we consider high-resolution cameras for the omnidirectional cameras can solve this problem. Another problem is the discrete omnidirectional models using 16 cameras. A better method is to acquire continuous models from the discrete observation and search by 2-D dynamic programming. We are now improving the algorithm.

The model, VAMBAM, is simple but general. Therefore, we consider this can be extended and realize more sophisticated recognition systems that can recognize more complex scenes, such as interactions among humans. Our final goal is to develop a *perceptual information infrastructure* to support human activities based on the proposed method.

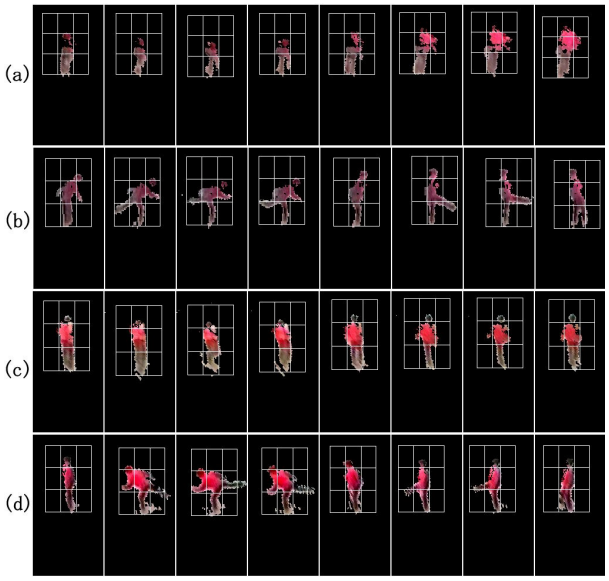


Fig. 16: An image sequence of "kick" gesture

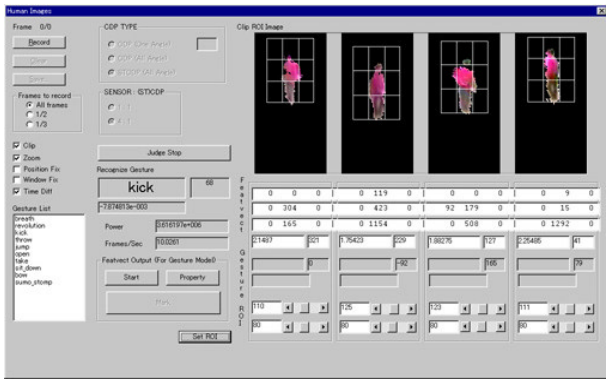


Fig. 17: Monitor output of the system

References

[Bobick, 1992] A. Bobick and R. C. Bolles, The representation space paradigm of concurrent evolving scene descriptions, IEEE Trans. PAMI, Vol. 14, No. 2, pp. 146-156, 1992.

[Ikeuchi, 1988] K. Ikeuchi, Automatic generation of object recognition program, Proc of IEEE, Vol. 76, No. 8, 1988.

[Ishiguro, 1997] H. Ishiguro, Distributed vision system: A perceptual information infrastructure for robot navigation, Proc. IJCAI, pp. 36-41, 1997.

[Ishiguro, 1998] H. Ishiguro, Development of low-cost and compact omnidirectional vision sensors and their applications, Proc. Int. Conf. Information systems, analysis and synthesis, pp. 433-439, 1998.

[Kato, 1999] K. Kato, H. Ishiguro and M. Barth, Identifying and localizing robots in a multi-robot system environment, Proc. IROS, 1999.

[Kitamura, 1990] Y. Kitamura and M. Yachida, Three-dimensional data acquisition by trinocular vision, Advanced Robotics, Vol.4, No.1, pp. 29-42, 1990.

[Koenderink, 1979] J.J. Koenderink and A.J. van Doorn, The internal representation of solid shape with respect to vision, Biological Cybernetics**,** 32:211-216, 1979

[Marr, 1982] D. Marr, Vision: A computational investigation into the human representation and processing of visual information, Freeman, W. H. and Company, 1982.

[Pavlovic, 1997] V.I. Pavlovic, R. Sharma and T.S. Huang, Visual Interpretation of hand gestures for human-computer interaction: A review, IEEE Trans. PAMI, vol. 19, no. 7, pp. 677-695, 1997.

[Sogo, 2000] T. Sogo, H. Ishiguro and M. Trivedi, Real-time target localization and tracking by N-ocular stereo, IEEE Workshop on Omni. Vision, pp. 153-160, 2000.

[Sogo, 2001] T. Sogo, H. Ishiguro and M. Trivedi, N-ocular stereo for real-time human tracking, In R. Benosman and S. B. Kang eds., Panoramic Vision: Sensors, Theory and Applications, Springer-Verlag, Berlin, 2001. (to appear).

Efficient Interpretation Policies

Ramana Isukapalli
101 Crawfords Corner Road
Lucent Technologies
Holmdel, NJ 07733 USA
ramana@research.bell-labs.com

Russell Greiner
Department of Computing Science
University of Alberta
Edmonton, AB T6G 2E8 Canada
greiner@cs.ualberta.ca

Abstract

Many imaging systems seek a good interpretation of the scene presented — *i.e.*, a plausible (perhaps optimal) mapping from aspects of the scene to real-world objects. This paper addresses the issue of finding such likely mappings *efficiently*. In general, an "(interpretation) policy" specifies when to apply which "imaging operators", which can range from low-level edge-detectors and region-growers through high-level token-combination–rules and expectation-driven object-detectors. Given the costs of these operators and the distribution of possible images, we can determine both the expected cost and expected accuracy of any such policy. Our task is to find a maximally effective policy — typically one with sufficient accuracy, whose cost is minimal. We explore this framework in several contexts, including the eigenface approach to face recognition. Our results show, in particular, that policies which select the operators that maximize *information gain per unit cost* work more effectively than other policies, including ones that, at each stage, simply try to establish the putative most-likely interpretation.

Keywords: vision, decision theory, real time systems

1 Introduction

Interpretation, *i.e.*, assigning semantically meaningful labels to relevant regions of an image, is the core process underlying a number of imaging tasks, including recognition ("is objectX in the image?") and identification ("which object is in the image?"), as well as several forms of tracking ("find all moving objects of typeX in this sequence of images"), etc. [PL95; HR96]. Of course, it is critical that interpretation systems be *accurate*. It is typically important that the interpretation process also be *fast*: For example, to work in real-time, an interpreter examining the frames of a motion picture will have only 1/24 of a second to produce an interpretation. Or consider a web-searcher that is asked to find images of, say, aircraft. Here again speed is critical — and most searchers do in fact sacrifice some accuracy to gain efficiency (*i.e.*, they quickly return a large number of "hits", only some of which are relevant). This paper addresses the challenge of producing an interpreter that is both sufficiently accurate and sufficiently efficient.

Section 2 provides the framework, showing how our framework generalizes the standard (classical) approaches to image interpretation, then providing a formal description of our task: given a distribution of possible images and an inventory of "operators", produce a "policy" that specifies when to apply which operator, towards optimizing some user-specified objective function. It describes three different policies that could be used, using a simple blocks-world example to illustrate these terms. The rest of this paper demonstrates that one of these policies, "INFOGAIN" (which uses information gain to myopically decide which operator is most useful at each step), is more effective than the other obvious contenders. Section 3 provides an empirical comparison of these approaches in the context of the simple blocks-world situation. Section 4 extends these ideas to deal with face recognition, using the modern eigenvector approach. (This complements Section 3's classical approach to interpretation.) Section 5 quickly surveys some related work. While the results in this paper demonstrate the potential for this approach, they are still fairly preliminary. Section 6 discusses some additional issues that have to be addressed towards scaling up this system.

2 Framework

2.1 Standard Approaches

There are many approaches to scene interpretation. A strictly "bottom-up" classical approach performs a series of passes over *all* of the information in the scene, perhaps going first from the pixels to edgels, then from edgels to lines, then to regions boundaries and then to descriptions, until finally producing a consistent interpretation for the entire scene. Most "top down", or "model driven", systems likewise begin by performing several bottom-up "sweeps" of the image — applying various low-level processes to the scene to produce an assortment of higher-level tokens, which are then combined to form some plausible hypothesis (*e.g.*, that the scene contains a person, etc.). These systems differ from strictly bottom-up schemes by then switching to a "top-down" mode: given sufficient evidence to support one interpretation, they seek scene elements that correspond to the parts of the proposed real-world object that have yet to be found [LAB90].

Notice the model-based systems have more prior knowledge of the scene contents than the strictly bottom-up schemes — in particular, they have some notion of "models" — which they can exploit to be more efficient. We propose going one step further, by using additional prior knowledge to further increase the efficiency of an interpretation system. Consider a trivial situation in which we only have to determine whether or not a "red fire-engine" appears in an image; and imagine, moreover, we knew that the *only* red object that might appear in our images is a fire-engine. Here it is clearly foolish to worry about line-detection or region-growing; it is

sufficient, instead, to simply sweep the image with an inexpensive "red" detector. Moreover, if we knew that the fire-engine would only appear in the bottom third of the image, we could apply this operator only to that region.

This illustrates the general idea of exploiting prior knowledge (*e.g.*, which objects are we seeking, as well as the distribution over the objects and views where they might appear) to produce an effective interpretation process. In general, we will assume our interpretation system also has access to the inventory of possible imaging operators. Given this collection of operators, an "interpretation policy" specifies when and how to apply which operator, to produce an appropriate interpretation of an image.

Our objective is to produce an *effective* interpretation policy — *e.g.*, one that *efficiently* returns a *sufficiently accurate* interpretation, where accuracy and efficiency are each measured with respect to the underlying task and the distribution of images that will be encountered. Such policies must, of course, specify the details: perhaps by specifying exactly which bottom-up operators to use, and over what portion of the image, if and when to switch from bottom-up to top-down, which aspects of the model to seek, etc. These policies can include "conditionals"; *e.g.*, "terminate on finding a red object in the scene; otherwise run procedure X". They may also specify applying a *particular* operator only to *specified* regions of the image (*e.g.*, seek only near-horizontal edges, only in the upper left quadrant of the image). Based on the information available, this interpretation policy could then use other operators, perhaps on other portions of the image to further combine the tokens found.

2.2 Input

We assume that our interpretation system "*IS*" is given the following information:

⋆ The **distribution** of images that the *IS* will encounter, encoded in terms of the distribution of objects and views that will be seen, etc. (Here we assume this information is explicitly given to the algorithm; we later consider acquiring this by sampling over a given set of training images.)

As a trivial example, we may know that each scene will contain exactly 25 sub-objects, each occupying a cell in a 5×5 grid; see Figure 1. Each of these cells has a specified "color", "texture" and "shape", and each of these properties ranges over 4 values. (Hence, we can identify each image with a $5 \times 5 \times 3 = 75$ tuple of values, where each value is from $\{1, 2, 3, 4\}$.) Moreover, our *IS* knows the distribution over these 4^{75} possible images; see below.

⋆ The **task** includes two parts: First, what objects the *IS* should seek, and what it should return — *e.g.*, "is there an airplane in image" or "is there a DC10 centered at $[43, 92]$ in the image", etc. *In our trivial blocks-world case, we simply want to know which of the images is being examined.*

Second, the task specification should also specify the "evaluation criteria" for any policy, which is based on both the expected "accuracy" and its expected "cost". In general, this will be a constrained optimization task, combining both hard constraints and an optimization criteria (*e.g.*, minimize some linear combination of accuracy and cost, or perhaps maximize the likelihood of a correct interpretation, for a given bound on the expected cost).

$c_{1,1}, t_{1,1}, s_{1,1}$	$c_{2,1}, t_{2,1}, s_{2,1}$			
		$c_{3,3}, t_{3,3}, s_{3,3}$		
				$c_{5,5}, t_{5,5}, s_{5,5}$

Figure 1: A simple image of 25 sub-objects

For this blocks-world task, we want to minimize the expected cost and also have 100% *correctness, assuming the operators are perfect.*

⋆ The **set of possible "operators"** includes (say) various edge detectors, region growers, graph matchers, etc. For each operator, we must specify

- its input and output, of the form: "given a set of pixel intensities, returns tokens representing the regions of the same color";
- its "effectiveness", which specifies the accuracy of the output, as a function of the input. This may be a simple "success probability", or could be of the form: "assuming noise-type W, can expect a certain ROC curve" [RH92];
- its "cost", as a function of (the size of) its input and parameter setting.

When used, each operator may be given some arguments, perhaps identifying the subregion of the image to consider.

Here, we consider three operators: O_1 *(resp.,* O_2*,* O_3*) for detecting the "value" of color (resp., "texture", "shape"); each mapping a* $[x, y]$ *location of the current image to a value in* $\{1, 2, 3, 4\}$*. (Note that location is an argument to the operator.) We assume that each operator, when pointed at a particular "cell", will reliably determine the actual value of that property at the specified location, and will do so with unit cost.* (Section 4 considers less trivial operators.)

For each situation, we assume our interpreter will be given a series of scenes, but will always have the same objective and criteria; *e.g.*, it is expected to look for the same objects in each image, and has a single objective function. (It is easy to generalize this to deal with environments that can ask different questions for different images, and impose different costs.)

At each stage, our *IS* will use its current knowledge (both prior information — *e.g.*, associated with the distribution and the operators — and information obtained by earlier probes) to decide whether (1) to terminate, returning some interpretation; or (2) to perform some operation, which involves specifying both the appropriate operator and the relevant arguments to this operator, and then recur.

2.3 Policies

In general, we could represent an *IS* explicitly as a large decision tree, whose leaf nodes each represent a complete interpretation (which is returned as the result of the *IS*), and whose internal nodes each correspond to a sequence of zero or more operator applications, followed by a test on the original data and/or some set of inferred tokens. Each arc descending from this node is labeled with a possible result of this test, and descends to new node (containing other operators and tests) appropriate for this outcome.

$IS_\chi(T = \langle C_{max}, P_{min} \rangle$: TaskSpecification,
$P(\,)$: Distribution, $O = \{o_i\}$: Operators, I: Image)
Initialize cost $C := 0$ Evidence $\vec{O} = \langle\rangle$
While [$C < C_{max}$ & $P_{min} > \max_x P(\text{Obj} = x \,|\, \vec{O})$] do
Select [o : operator; a : arguments] (based on policy χ, $P(\cdot)$)
(Note a may specify the region to consider)
Apply $o(a)$ to I, yielding v
Extend $\vec{O} := \vec{O} + \langle o(a), v \rangle$
Update $P(\cdot \,|\, \vec{O})$, based on result
Update $C := C +$ Cost[$o(a)$]
Return Best Interpretation: $\text{argmax}_x P(\text{Obj} = x \,|\, \vec{O})$

Figure 2: Identification algorithm,
for policy $\chi \in \{$RANDPOL, BESTHYP, INFOGAIN$\}$

Given that such explicit strategy-trees can be enormous, we instead represent strategies *implicitly*, in terms of a "*policy*" that specifies how to decide, at run-time, which operator to use. Figure 2 shows a general interpretation strategy using any of the policies. We will consider the following three policies:[1]

Policy RANDPOL**:** selects an operator[2] randomly.

Policy BESTHYP**:** first identifies the object that is most likely to be in the scene (given the evidence seen so far, weighted by the priors, etc.) and then selects the operator that can best verify this object [LHD+93, p370]: That is, after gathering information from k previous operators $\vec{O} = \langle o_1 = v_1, \ldots, o_k = v_k \rangle$, it computes the posterior probabilities of each possible interpretation s_i, $P(S = s_i \,|\, \vec{O})$. To select the next operator, BESTHYP will first determine which of the scenes is most likely — *i.e.*, $s^* = \text{argmax}_s \{P(S = s \,|\, \vec{O})\}$ — and then determines which operator has the potential of increasing the probability of this interpretation the most: Assume the operator o returns a value in $\{v_1, v_2 \ldots, v_j\}$; then o might increase the probability of s^* to best$(s^*, o) = \max_j \{P(S = s^* \,|\, \vec{O}, o = v_j)\}$. Here, BESTHYP will use the operator

$$o^*_{BH} \quad = \quad argmax_o \{ \text{ best}(s^*, o) \}$$

Policy INFOGAIN**:** selects the operator that provides the largest information gain (per unit cost) at each time. This policy computes, for each possible operator and argument combination o, the expected information gained by performing this operation: $EIG(S, \vec{O})o =$

$$H(S|\vec{O}) \; - \; \sum_j P(o = v_j \,|\, S, \vec{O}) \; H(S|\vec{O}, o = v_j)$$

[1]We view RANDPOL as a baseline; clearly we should not consider any system that does worse that this. These empirical studies are designed to test our hypothesis that INFOGAIN will actually work best in practice — and in particular, work better than BESTHYP, which is actually being used in some deployed applications [LHD+93].

[2]We will use the term "operator" to refer to the operator *instantiated with the relevant arguments*. Also, we will further abuse notation by writing $o_i = v_i$ to mean that the value v_i was obtained by applying the (instantiated) operator o_i to the image.

Figure 3: Tail lights of Chevrelot

where $H(S|E) = \sum_i P(S = s_i \,|\, E) \; \log P(S = s_i \,|\, E)$ is the entropy of the distribution over the interpretations S given the evidence E, for $E = \vec{O}$ or $E = \{\vec{O}, o = v_j\}$.

INFOGAIN then uses the operator

$$o^*_{IG} \quad = \quad argmax_o \{ \; EIG(\vec{O}, \, o) \; / C(o) \; \}$$

that maximizes $[EIG(\vec{O}, \, o)/C(o)]$, where $C(o)$ is the cost of applying the operator.

3 Simple Experiments: Blocks World

This section presents some experiments using the simple blocks world situation presented above. They are designed simply to illustrate the basic ideas, and to help us compare the three policies described earlier. Section 4 below considers a more realistic situation.

We first generated a set of 1000 images, each with 25 sub-objects, by uniformly assigning values for color, texture and shape to each of the sub-objects randomly, for each of 1000 images; we also assigned each a "prior distribution" p_i to these images (this corresponds to taking an empirical sample, with replacement). For each run, we randomly select one of the 1000 images to serve as a target for identification, then used each of the three policies to identify the image.

After observing $O_k = \{o_1 = v_1, o_2 = v_2, \ldots, o_k = v_k\}$ from the operators in the first k iterations, RANDPOL randomly selects a cell $C[i,j]_{RP}$ and an operator o_{RP} to probe the value for a property (color, texture etc), insisting only that o_{RP} was not tried earlier on $C[i,j]_{RP}$ in any previous iterations of this run. BESTHYP chooses a cell $C[i,j]_{BH}$ and an operator o_{BH} to maximize the posterior probability of the most likely image, as explained earlier. Finally, INFOGAIN chooses $C[i,j]_{IG}$ and o_{IG} such that $EIG(\vec{O}, o_{IG})/C(o_{IG})$ is the maximum over all possible cell and operator combinations. For each of these policies, the posterior probability is updated after applying the chosen operator on the cell. The process is repeated until the image is identified — *i.e.*, all other contenders are eliminated.

We considered 10 set-ups (each with its own objects and p_i's), and performed 5 runs for each set-up. Over these 50 runs, RANDPOL required on average 5.82 ± 0.27 probes, BESTHYP 5.44 ± 0.32 probes and INFOGAIN 5.32 ± 0.13. INFOGAIN is statistically better than the other two policies, at the $p < 0.1$ level.

We then performed a variety of other experiments in this domain, to help quantify the relative merits of the different policies — *e.g.*, in terms of the "average Hamming distances" between the images. See [IG01] for details.

While this particular task is quite simplistic, we were able to use the same ideas for the more interesting task of identifying the make and model of a car (*e.g.*, Toyota Corolla, Nissan Sentra, Honda Civic, etc.) given an image of the

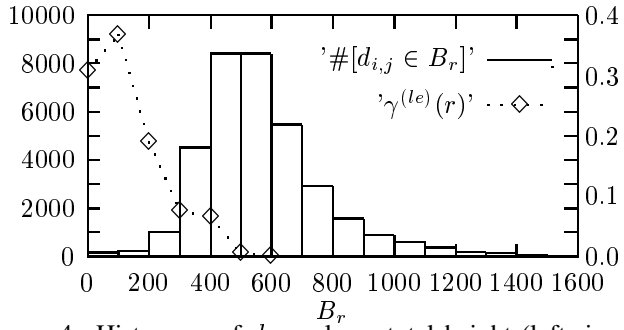


Figure 4: Histogram of $d_{i,j}$ values; total height (left-size scale) reflects number of $\langle i, j \rangle$ pairs in bucket B_r. (le feature; $k = 25$.) Also shows $\gamma^{(le)}(r)$, using right-size scale.

car that shows its "rear tail lights assembly"; see Figure 3. Again see [IG01] for details.

4 Scaling Up: Face Recognition

We next investigate the efficiency and accuracy of the three policies in the more complicated domain of "face recognition" [TP91; PMS94; PWHR98; EC97]. This section first discusses the prominent "eigenface" technique of face recognition that forms the basis of our approach; then presents our framework, describing the representation and the operators we use to identify faces; then presents our face interpretation algorithm; and finally shows our empirical results.

4.1 Eigenface, and EigenFeature, Method

Many of today's face recognition systems use *Principal Component Analysis* (PCA) [TP91]: Given a set of training images of faces, the system first forms the covariance matrix Σ of the images, then computes the k main eigenvectors of Σ, called "eigenfaces". Every training face h_i is then projected into this coordinate space ("facespace"), producing a vector, $\Omega_i = [\omega_1^{(i)}, \omega_2^{(i)}, \ldots, \omega_k^{(i)}]$.

During recognition, any test face h_{test} is similarly projected into the facespace, producing the vector Ω_{test}, which is then compared with each of the training faces. The best matching training face is taken to be the interpretation [TP91].

Following [PMS94], we extend this method to recognize facial features — eyes, nose, mouth, etc. — which we then use to help identify the individual in a given test image: We first partition the training data into two sets, $T = \{h_{1,i}\}$ (for constructing the eigenfeatures) and $S = \{h_{2,i}\}$ (for collecting statistics — see below), which each contains at least one face of each of the n people. Using $\mathrm{id}(h)$ to denote the person whose face is given by h, we have $\mathrm{id}(h_{1,i}) = i = \mathrm{id}(h_{2,i})$ for $i = 1..n$; each remaining $h_{1,j}$ and $h_{2,j}$ ($j > n$) also maps to $1..n$.

We use PCA on, say, the mouth regions of each T image, to produce a set of eigenvectors; here *eigen-mouths*. For each face image h_i, let $\Omega_i^{(m)}$ be "feature space" encoding of h_i's mouth-region. We will later compare the feature space encoding $\Omega_{test}^{(m)}$ of a new image h_{test} against these $\{\Omega_i^{(m)}\}$ vectors, with the assumption that $\Omega_{test}^{(m)} \approx \Omega_i^{(m)}$ suggests that h_{test} is really person i — *i.e.*, finding that $\|\Omega_{test}^{(m)} - \Omega_i^{(m)}\|$ is small should suggest that $\mathrm{id}(h_{test}) = i$. (Note $\|\cdot\|$ refers to the L_2 norm, aka Euclidean distance.)

Figure 5: Training images (top); test images (bottom)

To quantify how strong this belief should be, we compute $M = |T| \times |S|$ values $\{d_{i,j}\}$ where each $d_{i,j} = \|\Omega_{1,i}^{(m)} - \Omega_{2,j}^{(m)}\|$ is the Euclidean distance between the "eigen-mouth encodings" of T's $h_{1,i}$ and S's $h_{2,j}$. We considered 16 buckets $B_r \subset \Re$ for these $d_{i,j}$ values: $B_0 = [0, 100)$, $B_1 = [100, 200)$, ..., $B_{14} = [1400, 1500)$, $B_{15} = [1500, \infty)$. Then, for each bucket B_r, we estimate $P(\, d_{i,test} \in B_r \,|\, \mathrm{id}(h_{test}) = i \,)$ as

$$\gamma^{(m)}(r) \;=\; \frac{\#[\, d_{i,j} \in B_r \;\&\; \mathrm{id}(h_{1,i}) = \mathrm{id}(h_{2,j}) \,]}{\#[\, \mathrm{id}(h_{1,i}) = \mathrm{id}(h_{2,j}) \,]}$$

where $\#[\, \mathrm{id}(h_{1,i}) = \mathrm{id}(h_{2,j}) \,]$ is the number of $\langle i, j \rangle$ pairs where $h_{1,i} \in T$ is the same person as $h_{2,j} \in S$. (We used the obvious Laplacian correction to avoid using 0s here [Mit97].) We also compute

$$\rho^{(m)}(r) \;=\; \frac{\#[\, d_{i,j} \in B_r \,]}{|S| \times |T|}$$

to estimate $P(\, d_{i,test} \in B_r \,)$. (Note this is the average over *all* images i in the test set T.) Figure 4 shows a histogram of the $d_{i,j}$ values, using the 16 buckets for the left eye feature (see below); it also shows the values of $\gamma^{(le)}(r)$ for $r = 0..6$.

We use these $\{\gamma^{(m)}(r)\}_r$ and $\{\rho^{(m)}(r)\}_r$ values to interpret a new test image of a person's face h_{test}. (While the specific image h_{test} is not in $T \cup S$, it is another face of someone who has other faces in $T \cup S$).) We first project h_{test}'s mouth region onto the "eigen-mouth"s space, forming the vector $\Omega_{test}^{(m)}$, then compare $\Omega_{test}^{(m)}$ with the stored eigen-mouth projections (from T) — computing the values $d_{i,test} = \|\Omega_i^{(m)} - \Omega_{test}^{(m)}\|$ for each i in T. This $d_{i,test}$ will be in some bucket, say $B_r = [r, r + 100)$. We then use Bayes Rule to compute the probability that this face is person i:

$$\begin{aligned} & P(\, \mathrm{id}(h_{test}) = i \,|\, \Omega_{test}^{(m)}, \{\Omega_j^{(m)}\}_j \,) \\ &= P(\, \mathrm{id}(h_{test}) = i \,|\, d_{i,test} \in B_r \,) \\ &= \frac{P(\, d_{i,test} \in B_r \,|\, \mathrm{id}(h_{test}) = i \,)\; P(\, \mathrm{id}(h_{test}) = i \,)}{P(\, d_{i,test} \in B_r \,)} \\ &\approx \gamma^{(m)}(r) \times \tfrac{1}{n} / \rho^{(m)}(r) \end{aligned} \tag{1}$$

(Here, we assume the faces are drawn from the n individuals uniformly; hence $P(\, \mathrm{id}(h_{test}) = i \,) = 1/n$.)

So far, we considered only a *single* feature — here, "mouth projections", as indicated by the (m) superscript. We similarly compute $\gamma^{(n)}(r)$, $\gamma^{(le)}(r)$, $\gamma^{(re)}(r)$, values associated with the nose, left-eye and right-eye, as well as the $\rho^{(n)}(r)$, $\rho^{(le)}(r)$, $\rho^{(re)}(r)$ values.

We then used the Naïve-Bayes assumption [DH73] (that features are independent, given the specified person) to essentially simply multiply the associated probabilities: Assume we observed, for each feature f, $\|\Omega^{(f)}_{test} - \Omega^{(f)}_{i}\| \in B_{r_f}$, then

$$\begin{aligned} P(\,\mathrm{id}(h_{test}) = i \,|\, \Omega^{(m)}_{test},\, \Omega^{(le)}_{test},\, \Omega^{(n)}_{test},\, \{\Omega^{(f)}_{i}\}_{f,i}\,) \\ = \;\alpha \;\cdot\; P(\,\Omega^{(m)}_{test}, \Omega^{(le)}_{test}, \Omega^{(n)}_{test} \,|\, \mathrm{id}(h_{test}) = i\,) \\ = \;\alpha \;\cdot P(\,\Omega^{(m)}_{test} \,|\, \mathrm{id}(h_{test}) = i\,) \;\cdot \\ P(\,\Omega^{(le)}_{test} \,|\, \mathrm{id}(h_{test}) = i\,) \cdot P(\,\Omega^{(n)}_{test} \,|\, \mathrm{id}(h_{test}) = i\,) \\ \approx \quad \alpha' \cdot \frac{\gamma^{(m)}(r_m)}{\rho^{(m)}(r_m)} \cdot \frac{\gamma^{(le)}(r_{le})}{\rho^{(le)}(r_{le})} \cdot \frac{\gamma^{(n)}(r_n)}{\rho^{(n)}(r_n)} \qquad (2) \end{aligned}$$

where α, α' are scaling constants (as the prior is uniform).

(Note: we had also tried computing individual $\gamma^{(m)}_i(r)$ values, specific to each training face $h_{1,i}$. However, we found this was too noisy, as the number of relevant instances was too small.)

4.2 Framework

The **distribution** is the set of all people who can be seen, which varies over race, gender and age, as well as poses and sizes; we approximate this using the images given in the training set. We assume that any test face-image belongs to one of the people in the training set, but probably with a different facial expression or in a slightly different view, and perhaps with some external features not in the training image (like glasses, hat, etc.), or vice versa. Figure 5 shows three training images (top) and four test images (bottom).

Our **task** is to identify the person from his/her given test image h_{test} (wrt the people included in the training set), subject to the minimum acceptable accuracy (P_{min}) and the maximum total cost of identification (C_{max}).

We use four classes of **operators**, $O = \{\, o_{le}(k),\;\; o_{re}(k),\;\; o_{n}(k),\;\; o_{m}(k)\,\}$ to detect respectively "left eye", "right eye", "nose" and "mouth". Each specific operator also takes a parameter k which specifies the size of the feature space to consider; here we consider $k \in \{25, 30, 35, 40, 45\}$. As discussed above, each instantiated operator $o \in O$ takes an input the test image of a face h_{test}, and returns a probabilistic distribution over the individuals.

Each operator $o_f(k)$ (associated with the feature $f \in \{le,\; re,\; n,\; m\}$) performs three subtasks: SubTask#1 locates the feature f_{test} from within the entire face h_{test}. Here we use a simple template matching technique in which we search in a fixed "window" of size $P \times Q$ pixels in h_{test} for any given feature, of size $p < P$ by $q < Q$ pixels. SubTask#2 then projects the relevant region f_{test} of the test image into the feature space — computing $\Omega^{(f)}_{test}$ of dimension k. SubTask#3 uses this $\Omega^{(f)}_{test}$ to compute first the values $d_{i,test} = \|\Omega^{(f)}_{i} - \Omega^{(f)}_{test}\|$ for each person i, then place each $d_{i,test}$ value into the appropriate B_r bucket, and finally compute the probability $P(\,\mathrm{id}(h_{test}) = i \,|\, \Omega^{(f)}_{test}\,)$ for each person i, using Equation 1 (possibly augmented with Equation 2 to update the distribution when considering the 2nd and subsequent features); see Section 4.1. For each eigenspace dimension k, we empirically identified the cost (in seconds) of the four classes of operators — $C(o_{le}(k)) = 0.65 + (k-25)\times 0.021$, $C(o_{re}(k)) = 0.87 + (k-25)\times 0.015$, $C(o_n(k)) = 1.54 + (k-25)\times 0.025$ and $C(o_m(k)) = 1.23 + (k-25)\times 0.04$. While increasing the dimensionality k of the feature space should improve the accuracy of the result, here we see explicitly how this will also increase the cost.

4.3 Interpretation Phase

During interpretation, each current policy (RANDPOL, BESTHYP and INFOGAIN) iteratively selects an operator $o(k) \in O$. RANDPOL chooses an operator o_{RP} and a value k_{RP} randomly, subject only to the condition that o_{RP} had not been tried before on the image; BESTHYP first identifies the most likely person $i = \mathrm{argmax}\, P(\,\mathrm{id}(h) = i \,|\, \cdot\,)$, then choses the instantiated $o_{BH}(k_{BH})$ operator that best confirms this hypothesis (that the image belongs to i person), provided this o_{BH} had not been used earlier for this image; and INFOGAIN chooses an instantiated operator $o = o_{IG}(k_{IG})$ that has the maximum $EIG(\cdot, o)/C(o))$ value. In each case, the operator is applied to the appropriate region in the given test face image and the distribution is updated... until one face is identified with sufficiently high probability ($> P_{min}$) or the system fails (by exhausting all the possible operators, or having cost $> C_{max}$); see Figure 2.

4.4 Face Recognition Experiments

We used 534 face images of 102 different people, each 92×112 pixels, from which we placed 187 images into T, another 187 images into S (T and S are used in the training phase to collect statistics) and used the remaining 260 as test images. As shown in Figure 5, the faces are more or less in the same pose (facing front), with some small variation in size and orientation.[3] We considered all 20 operators based on the four features listed above and $k \in \{25, 30, 35, 40, 45\}$ for each feature.

Basic Experiment: We set $P_{min} = 0.9$ and $C_{max} = \infty$ (*i.e.*, no upper limit on identification cost). In each "set-up", we assigned a random probability to each person. On each run, we picked one face randomly from the test set as the target, and identified it using each of the three policies. We repeated this for a total of 25 runs per set-up, then changed the probability distribution and repeated the entire process again, for a total of 8 set-ups. The cost of recognition on the average was 8.764 ± 0.586, 7.674 ± 0.702 and 6.811 ± 0.702

[3] (1) All these faces were downloaded from the web sites `whitechapel.media.mit.edu` and `www.cam-orl.co.uk`. (2) This work assumes the head has already been located and normalized; if not, we can use standard techniques [TP91] first. (3) All the experiments reported in this paper were run on a Pentium 200 MHz. PC with 32 MB. RAM running Linux 2.0.35 OS.

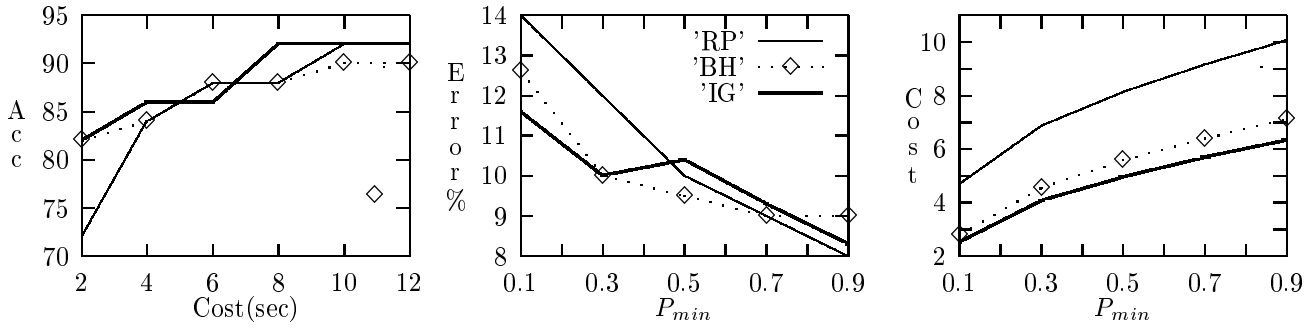


Figure 6: (a) Cost vs. Accuracy (b) Min. Accuracy vs. Error (c) Min. Accuracy vs. Cost

seconds for RANDPOL, BESTHYP and INFOGAIN respectively. INFOGAIN is statistically better than the other two policies, at the $p < 0.1$ level. (As expected, these policies had comparable identification accuracy here: 89.16%, 90.84% and 90.84%, respectively.)

Bounding the Cost: In many situations, we need to impose a hard restriction on the total cost; we therefore considered $C_{max} \in \{2, 4, \ldots, 12\}$ seconds. We then picked one face randomly from the test set, and identified the test image for each of these maximal costs, using each of the three policies. As always, we terminate whenever the probability of any person exceeds P_{min} or if the cost exceeds C_{max}, and return the most likely interpretation.

We repeated this experiment for a total of 10 set-ups (each with a different distribution over the people) and with 25 random runs (target face images) per set-up. The accuracy (the percentage of correct identifications) for each policy is shown for various values of C_{max} in Figure 6(a). INFOGAIN has better accuracy than both BESTHYP and RANDPOL. RANDPOL trailed these two policies significantly for low (≤ 4 seconds) cost.

Varying the Minimum Accuracy: In this experiment, we varied P_{min} from 0.1 to 0.9. For each of these values, we chose a face randomly from the test set as the target and identified it using each of the three policies. During the process, the first person i in the training set for which $P(\,\mathrm{id}(h) = i \,|\, \cdot\,) > P_{min}$ is returned (or if cost $> C_{max}$, the most probable face is returned). We repeated this for 25 different target faces (runs) per set-up, and repeated the entire process for a total of 8 different set-ups.

We evaluated the results in two different ways. First, Figure 6(b) compares the percentage of wrong identifications of each policy, for each P_{min} value. INFOGAIN has fewer wrong identifications than BESTHYP and RANDPOL for low accuracy. As expected, for sufficiently high accuracy, all three policies have comparable number of wrong identifications. Secondly, Figure 6(c) compares the average cost of each policy, for each P_{min} value. Again, INFOGAIN has lower cost than BESTHYP and RANDPOL, while RANDPOL trails the other two policies significantly.

5 Literature Survey

Our work formally investigates the use of decision theory in image interpretation, explicitly addressing accuracy versus efficiency tradeoffs [GE91; RSP93]. Geman and Jedynak [GJ96] used information theoretic approaches to find the "optimal sequence of questions" for recognizing objects — hand-written numerals (0-9), and highways from satellite images. Our work also uses information theoretic methods to compute the expected information gain, but we differ as we are seeking the most cost-effective sequence of interpretation operators, rather than the shortest sequence of questions; this forces us to explicitly address the cost vs accuracy tradeoffs. Sengupta and Boyer [SB93] presented a hierarchically structured approach to organizing large structural model-bases using an information theoretic criterion. We do not have explicit model-bases, but consider various interpretation policies that decide, at run time, which operators to apply to what regions of an image, based on expected information gain of the operators.

We used the domain of face recognition to test our approach. While there are several approaches to face recognition [TP91; PMS94; PWHR98; EC97], none *explicitly* address the issues of efficiency. We used one of the popular and successful methods as the basis to our approach and performed a systematic study of the various efficiency and accuracy related issues.

Levitt et al. [LAB90; BLM89; LHD+93] have applied Bayesian inference methods and influence diagrams to image interpretation. We however provide a way to adjust the optimization function. (Our work also further motivates the use of "maximum expected utility" in such systems.)

As our system is seeking a policy that maps the state (here current distribution over possible interpretations) to an appropriate action, it can be viewed as solving a Markov decision problem (MDP), which puts it in the realm of reinforcement learning; *cf.*, [Dra96]. Our research objective differs as we are considering a range of reward functions, which can be various combinations of accuracy and efficiency (some of which may be difficult to word within a MDP framework). We anticipate being able to use many of the Reinforcement Learning techniques as we begin to consider interactions between the actions, and going beyond our current myopic approach.

Finally, there is a growing body of work on providing precise characteristics of various imaging operators, which quantify how they should work [Har94; RH92]. We hope to use these results to quantify the effectiveness of our operators, to help our algorithms decide when to use each. There is also work on building platforms that allow a user to *manually* assemble these operators [Fua97; PL94], often using an expert-system style approach [Mat89].

Here, we are taking a step towards automating this process, wrt some given task. In particular, our approach suggests a way to *automatically* assemble the appropriate imaging operators (*i.e.*, without human intervention), as required to effectively interpret a range of images.

6 Conclusions

Future Work: While our face recognition results show that our ideas can be applied to a complex domain, there are a number of extensions that would further scale up our approach. Some are relatively straightforward — *e.g.*, extending the set of operators to cover more features; this will help deal with larger number of faces in the training set, with better accuracy and lower interpretation costs. In other contexts, we will need to deal with thornier issues, such as operators that rely on one another. This may be because one operator requires, as input, the output of another operator (*e.g.*, a line-segmenter produces a set of tokens, which are then used by a line-grower — notice this precondition-situation leads to various planning issues [CFML98]), or because the actual data obtained from one operator may be critical in deciding which next operator (or parameter setting) to consider next: *e.g.*, finding the fuselage at some position helps determine where to look for the airplane's wings.

Clearly we will need to re-think our current myopic approach to cope with these multi-step issues; especially as we expect heuristics will be essential, as this task is clearly NP-hard [Sri95]. Finally, all of this assumes we have the required distribution information. An important challenge is more efficient ways to acquire such information from a set of training images — perhaps using something like Q-learning [SB98].

Contributions: This paper has three main contributions: First, it provides a formal foundation for investigating *efficient* image interpretation, by outlining the criteria to consider, and suggesting some approaches. Secondly, our implementation is a step towards *automating* the construction of effective image interpretation systems, as it will automatically decide on the appropriate policies for operator applications, as a function of the user's (explicitly provided) task and the available inventory of operators. Finally, it presents some results related to these approaches — in particular, our results confirm the obvious point that information gain (as embodied in the INFOGAIN policy) is clearly the appropriate measure to use here — and in particular, it is better than the BESTHYP approach. This observation is useful, as there are deployed imaging systems that use this BESTHYP approach [LHD+93].

Acknowledgements

RG gratefully acknowledges support from NSERC for this project. RI thanks Sastry Isukapalli for several discussions, general comments and help on the paper. Both authors thank Ramesh Visvanathan, and the anonymous reviewers, for their many helpful and insightful comments.

References

[BLM89] T Binford, T Levitt, and W Mann. Bayesian inference in model based machine vision. In *UAI*, 1989.

[CFML98] S Chien, F Fisher, H Mortensen, and E Lo. Using ai planning techniques to automatically reconfigure software modules. In *Lecture Notes in CS*, 1998.

[DH73] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

[Dra96] B. Draper. Learning control strategies for object recognition. In Ikeuchi and Veloso, editors, *Symbolic Visual Learning*. Oxford University Press, 1996.

[EC97] K Etemad and R Chellappa. Discriminant analysis for recognition of human faces. *J. Optical Society of America*, 1997.

[Fua97] P Fua. *Image Understanding for Intelligence Imagery*, chapter Model-Based Optimization: An Approach to Fast, Accurate, and Consistent Site Modeling from Imagery. Morgan Kaufmann, 1997.

[GE91] R. Greiner and C. Elkan. Measuring and improving the effectiveness of representations. In *IJCAI91*, pages 518–524, August 1991.

[GJ96] G Geman and B Jedynak. An active testing model for tracking roads in satellite images. *IEEE PAMI*, 1996.

[Har94] R Haralick. Overview: Computer vision performance characterization. In *ARPA IU*, 1994.

[HR96] R Huang and S Russell. Object identification: A bayesian analysis with application to traffic surveillance. *Artificial Intelligence*, 1996.

[IG01] R. Isukapalli and R. Greiner. Efficient car recognition policies. In *ICRA*, 2001.

[LAB90] T S Levitt, J M Agosta, and T O Binford. Model-based influence diagrams for machine vision. In *UAI*, 1990.

[LHD+93] T. Levitt, M. Hedgecock, J. Dye, S. Johnston, V. Shadle, and D. Vosky. Bayesian inference for model-based segmentation of computed radiographs of the hand. *Artificial Intelligence in Medicine*, 1993.

[Mat89] T. Matsuyama. Expert systems for image processing: knowledge-based composition of image analysis processes. *Computer Vision, Graphics, and Image Processing*, 48(1):22–49, 1989.

[Mit97] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[PL94] A. Pope and D. Lowe. Vista: A software environment for computer vision research. In *IEEE CVPR*, 1994.

[PL95] A. Pope and D. Lowe. Learning object recognition models from images. In *Early Visual Learning*, 1995.

[PMS94] A Pentland, B Moghaddam, and T Starner. View-based and modular eigenspaces for face recognition. In *IEEE CVPR*, 1994.

[PWHR98] P Phillips, H Wechsler, J Huang, and P Rauss. The feret database and evaluation procedure for face recognition algorithms. *Image and Vision Computing*, 1998.

[RH92] V Ramesh and R M Haralick. Performance characterization of edge operators. In *Machine Vision and Robotics Conference*, 1992.

[RSP93] S. Russell, D. Subramanian, and R. Parr. Provably bounded optimal agents. In *IJCAI93*, August 1993.

[SB93] K Sengupta and K Boyer. Information theoretic clustering of large structural modelbases. In *IEEE CVPR*, 1993.

[SB98] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[Sri95] S Srinivas. A polynomial algorithm for computing the optimal repair strategy in a system with independent component failures. In *UAI*, 1995.

[TP91] M Turk and A Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 1991.

ROBOTICS AND PERCEPTION

VISION II

Perceptual Texture Space Improves Perceptual Consistency of Computational Features

Huizhong Long and Wee Kheng Leow
School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543
longhuiz, leowwk@comp.nus.edu.sg *

Abstract

Perceptual consistency is important in many computer vision applications. Unfortunately, except for color, computational features and similarity measurements for other visual features are not necessarily consistent with human's perception. This paper addresses three critical issues regarding perceptually consistent texture analysis: (1) development of perceptual texture space, (2) assessment of how consistent computational features are to human perception, and (3) mapping computational features to perceptual space. It demonstrates the construction of a reliable perceptual texture space, which can be used as a yardstick for assessing the perceptual consistency of computational features and similarity measurements. Moreover, it is found that commonly used computational texture features are not very consistent with human perception, and mapping them to the perceptual space improves their perceptual consistency.

1 Introduction

Perceptual consistency is important in many computer vision applications. For example, a computer image understanding system should analyze and interpret an image in a manner that is consistent with human's perception of the image. A content-based image retrieval system should compare images in its database with the query in a manner that is consistent with human's perception of image similarity. To this end, many systems and algorithms measure color similarity in the CIE L*u*v* space which has been shown to be quite consistent with human's perception [Meyer and Greenberg, 1987]. Unfortunately, computational features and similarity measurements for other visual features are not necessarily consistent with human's perception, with the possible exception of Santini and Jain's *fuzzy features contrast model* [Santini and Jain, 1999].

Several perceptual spaces that capture human's perception of texture exist. However, the mapping from computational features to these perceptual spaces has not been derived. Another difficulty with using these perceptual spaces is that the scales and orientations of the textures used to construct the spaces are not normalized. Therefore, texture similarity measured in these spaces are confounded by scale and orientation variations. This problem makes it very difficult to use existing texture spaces in practical applications.

This paper addresses three critical issues regarding perceptually consistent image analysis, with specific application to texture analysis:

1. Development of a *perceptual texture space* (PTS) based on texture images with canonical scales and orientations (Section 3). Once the reliability of PTS is established, it can be used as a yardstick for assessing the perceptual consistency of computational features and similarity measurements.
2. Assessment of how consistent commonly used computational texture features and similarity measurements are to human's perception (Section 4).
3. Assessment of the accuracy of mapping computational texture features to PTS (Section 5). Once an accurate mapping is accomplished, texture similarity can be measured in PTS to achieve perceptual consistency.

2 Perceptual Consistency

There are many ways of defining perceptual consistency. This section discusses some definitions which will be used throughout the paper. Let p_{ij} denote the perceptual distance between samples i and j, and d_{ij} denote the corresponding measured or computational distance. A simple notion of perceptual consistency is that d_{ij} is *proportional* to p_{ij}. That is, there exists a linear function f such that

$$p_{ij} = f(d_{ij}), \quad \forall i, j. \tag{1}$$

Then, perceptual consistency can be measured in terms of the mean squared error (MSE) e of linear regression:

$$e = \frac{1}{N} \sum_{i,j} (p_{ij} - f(d_{ij}))^2 \tag{2}$$

where N is the number of sample pairs. The smaller the MSE, the better is the consistency. A perfect consistency has an MSE of 0.

*This research is supported by NUS ARF R-252-000-049-112 and joint NSTB and NUS ARF RP3999903.

A less stringent notion of perceptual consistency is to require that f be a monotonic function which can be nonlinear. The difficulty with this definition is that it is difficult to determine the best nonlinear function to use in practice.

An alternative definition is to require that d_{ij} be *statistically correlated* to p_{ij}. In this case, it is useful to transform the populations $\{d_{ij}\}$ and $\{p_{ij}\}$ to equivalent zero-mean unit-variance populations $\{d'_{ij}\}$ and $\{p'_{ij}\}$:

$$p'_{ij} = \frac{p_{ij} - \bar{p}}{\sigma_p}, \quad d'_{ij} = \frac{d_{ij} - \bar{d}}{\sigma_d} \tag{3}$$

where $\bar{p}$ and $\bar{d}$ are the means and σ_p and σ_d are the standard deviations of the populations. Then, perceptual consistency can be measured in terms of the correlation r:

$$r = \frac{1}{N} \sum_{i,j} p'_{ij}\, d'_{ij} \; . \tag{4}$$

Substituting Eq. 3 into Eq. 4 yields the Pearson's correlation coefficient:

$$r = \frac{\sum_{i,j} (p_{ij} - \bar{p})(d_{ij} - \bar{d})}{\left[\sum_{i,j} (p_{ij} - \bar{p})^2 \sum_{i,j} (d_{ij} - \bar{d})^2 \right]^{1/2}} \; . \tag{5}$$

The coefficient r ranges from -1 to $+1$ (for perfect correlation). In the following sections, we will use both MSE and Pearson's correlation coefficient to measure two different aspects of perceptual consistency. Note that, with perfect consistency ($e = 0$ or $r = 1$), we obtain the following condition:

$$d_{ij} \leq d_{kl} \Rightarrow p_{ij} \leq p_{kl} \;\; \text{for any textures } t_i, t_j, t_k, t_l. \tag{6}$$

That is, if perfect consistency is achieved, computational similarity would imply perceptual similarity. This condition is especially useful in practical applications (Section 6).

3 Development of Perceptual Texture Space

This section addresses the first critical issue put forth in Section 1: development of perceptual texture space (PTS). First, let us briefly review existing perceptual texture models.

3.1 Existing Perceptual Texture Models

The earliest study of human's perception of texture similarity was conducted by Tamura et al. [Tamura *et al.*, 1978] In their experiments, 48 human subjects were asked to judge the similarity of texture pairs according to six visual properties, namely, coarseness, contrast, directionality, line-likeness, regularity, and roughness. Similarity judgments were measured and each texture was assigned a perceptual rating value along each of the six visual scales. Due to the combinatorial nature of the task, only 16 textures were used. Amadasun and King [Amadasun and King, 1989] conducted similar ranking experiments to measure similarity judgments according to various visual properties.

The major difficulty with these studies is that the subjects were asked to judge texture similarity according to subjective visual properties. Unfortunately, the subjects' interpretations of the meaning of these visual properties are expected to vary from one person to the next. Therefore, it is uncertain whether the individual ranking results can be combined into group ranking results that represent the perception of a typical person. The second difficulty is that the ranking results were measured according to individual visual properties. But, the relative scale between two visual properties is unknown. For example, one unit difference in coarseness may not be perceptually equal to one unit difference in regularity. So, the different visual dimensions cannot be easily combined to form a perceptual texture space.

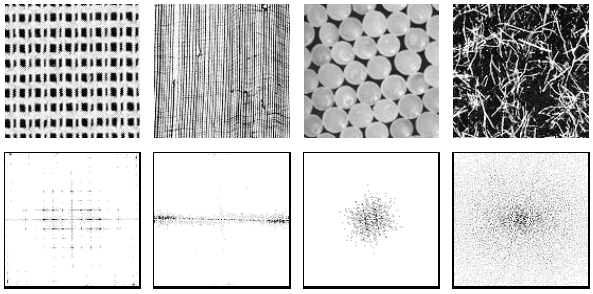
To avoid these difficulties, Rao and Lohse [Rao and Lohse., 1993] performed an experiment in which 20 subjects were asked to sort 56 textures into as many groups as the subjects wished such that the textures in each group were perceptually similar. The textures were sorted based on the subjects' perception of overall texture similarity instead of individual visual properties. A co-occurrence matrix of the sorting results was computed and multidimensional scaling [Hair *et al.*, 1998] was performed to derive a 3D perceptual space. Rao and Lohse concluded that the 3 dimensions of the space strongly correlate with the visual properties of repetitiveness, orientation, and complexity.

Heaps and Handel [Heaps and Handel, 1999] conducted further studies using the same methodology. However, they arrived at different conclusions than those of Rao and Lohse. They concluded that it is not possible to reliably associate a single visual property to each dimension of the texture space because each dimension appears to be correlated with a combination of more than one visual property. In addition, perception of texture similarity depends on the context in which the similarity is judged. That is, how similar two textures appear to humans depends not only on the two textures being judged, but also on the whole set of textures with which pairwise judgments are made.

In our development of perceptual texture space, we do not assign visual properties to the dimensions of the space. Moreover, the influence of the context problem is reduced by (1) selecting most, if not all, texture patterns that are expected to be present in images of natural scenes, and (2) normalizing the intensity, contrast, scale, and orientation of the textures used in the psychological experiment. It is well known, for example, that human's sensitivity of perceiving spatial frequency (i.e., spatial patterns) is dependent on contrast [Schiffman, 1996]. Therefore, these confounding factors should be removed from the experiment.

3.2 Texture Image Preparation

A rich set of texture images are collected to construct the perceptual texture space (PTS). Fifty texture images are selected from the Brodatz album. The remaining images in the album are omitted for various reasons. For example, some textures are scaled versions of each others, and only one of them is required. Another reason is that the contents of some images (e.g., flowers in lace) appear very striking but their texture patterns are perceptually weak. It is very difficult for the subjects to avoid classifying the images based on the content instead of the texture alone. In addition to these Brodatz textures, ten images of natural textures such as foliage and roof tiles are included because they are not collected in the Brodatz



(a) (b) (c) (d)

Figure 1: Four types of DFT patterns (bottom) and the corresponding representative textures (top): (a) structured, (b) directional, (c) granular, (d) random.

album. These 60 texture images constitute a good set of representative textures that appear in images of natural scenes. These images are first normalized before they are used in the psychological experiments.

Four image features, namely, intensity, contrast, scale, and orientation, that can confound the subject's judgment are normalized. The intensity and contrast of a texture image is measured as the mean and standard deviation of the intensity of the pixels in the image. A texture's intensity and contrast are normalized by a simple linear transformation to a canonical mean intensity and standard deviation.

Our analysis reveals that there is no single scaling method that fits all the textures. However, by categorizing the textures into four different groups based on their 2D discrete Fourier transform (DFT) patterns, each group of textures can be scaled in the same way. The four groups consist of structured, directional, granular, and random textures (Fig. 1).

Structured textures have very well defined DFT patterns which contain a small number of highly localized, prominent frequency components with large amplitudes (Fig. 1a). The textures in this group are scaled and rotated so that the frequencies of the 8 most prominent components in the DFTs of various textures are identical. The DFT patterns of oriented textures have many frequency components and well defined orientations (Fig. 1b). They lack prominent frequency components which can be used for scale normalization. However, their orientation can be normalized so that their DFT patterns have the same orientations.

Granular textures have well defined frequencies and many orientations. Their DFT patterns appear in the form of small disks (Fig. 1c). Their scales are normalized so that their DFT patterns have approximately the same size. Random textures have many frequencies and orientations (Fig. 1d). They lack well defined scales and orientations. So, they are not subjected to scale and orientation normalization.

3.3 Measurement of Human Perception

A direct way of measuring the dissimilarity δ_{ij} between textures t_i and t_j is to perform an exhaustive comparison of all $n(n-1)/2$ sample pairs. When the number of samples n is large (e.g. 60), the number of pairs becomes very large (in this case, 1770). It is practically impossible for a subject to perform such a large number of judgments. Another commonly used method is to sort the samples into as many groups as the subjects wish such that the samples in each group are perceptually similar. This method sorts n samples into g groups, requiring at most ng judgments, which is far smaller than $n(n-1)/2$. This method is also adopted in [Rao and Lohse., 1993] and [Heaps and Handel, 1999].

In our experiment, sixty subjects were asked to freely sort the images according to texture similarity. The sorting results were used to construct distance matrices based on two similarity measurements: co-occurrence matrix and information measurement [Donderi, 1988].

Let c_{ij} denote the number of times textures t_i and t_j are grouped into the same group. Then, the distance d_{ij} measured by the co-occurrence method is:

$$d_{ij} = n - c_{ij} \tag{7}$$

where n is the number of subjects in the experiment.

Donderi's distance measurement d_{ij} is based on Shannon's information theory :

$$d_{ij} = \sqrt{H(t_i|t_j) + H(t_j|t_i)} \tag{8}$$

where $H(t_i|t_j)$ is the *conditional entropy* or *independent information* of t_i after knowing the grouping of t_j. Let G denote the set of textures used in the experiment and $G(t_i)$ denote the group in which t_i belongs. Then, the independent information $H(t_i|t_j)$ is computed as follows:

$$H(t_i|t_j) = \begin{cases} 0 & \text{if } G(t_i) = G(t_j) \\ \log_2 |G| - \log_2 |G(t_i)| & \text{if } G(t_i) \neq G(t_j) \end{cases}. \tag{9}$$

According to Eq. 8, informationally independent stimuli are dissimilar and informationally redundant stimuli are similar. It has been demonstrated that information measurements derived from the free sorting task are empirically and theoretically equivalent to the distance measurements generated by direct pairwise comparison [Donderi, 1988].

3.4 Construction of Perceptual Space

After obtaining the distance matrices, multidimensional scaling (MDS) was applied to derive a perceptual texture space (PTS). MDS, also known as *perceptual mapping*, maps the original dissimilarity judgments δ_{ij} into points in a multidimensional space such that each point corresponds to a texture, and the Euclidean distance p_{ij} between two textures t_i and t_j matches as well as possible the original dissimilarity δ_{ij}. The multidimensional space would correspond to a perceptual space, and the Euclidean distance measured in the space would correspond to perceptual distance.

The appropriate number of dimensions of the perceptual space is determined by the *stress* measure which indicates the proportion of the variance of the data that is not accounted for [Hair *et al.*, 1998]:

$$\text{stress} = \sqrt{\sum_{i,j} (p_{ij} - \delta_{ij})^2 \Big/ \sum_{i,j} p_{ij}^2} \tag{10}$$

The larger the number of dimensions, the better is the fitting of the data and the lower is the stress. However, a space

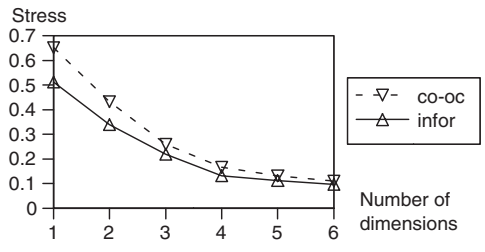


Figure 2: Stress plot of the MDS fitting results. Optimal trade-off between accurate fitting and small number of dimensions is obtained with four dimensions. The space constructed with Donderi's information measurement (infor) has a better fit than that constructed using co-occurrence (co-oc).

Table 1: Comparison of Heaps and Handels's space and our perceptual texture space (PTS) with Rao and Lohse's space. Pearson's correlation coefficients show that the spaces are consistent with each others.

perceptual space	3D	4D	5D
Heaps & Handel	0.790	–	–
PTS (co-occurrence)	0.722	0.732	0.713
PTS (info. measure)	0.726	0.694	0.695

with too many dimensions is difficult to visualize and to use. So, a trade-off is made to find a small number of dimensions that offer good data fitting. Rao and Lohse [Rao and Lohse., 1993], as well as Heaps and Handel [Heaps and Handel, 1999], concluded that a texture space with three dimensions is appropriate. Our MDS fitting results show that increasing the number of dimensions beyond 4 does not significantly reduce the stress. Thus, a four-dimensional texture space provides a better trade-off (Fig. 2).

The constructed spaces are verified by comparing them with existing perceptual spaces. This comparison is performed by computing the Pearson's correlation (Eq. 5) between the distances measured in the spaces. Table 1 summarizes the comparison results. Heaps and Handel reported a good correlation ($r = 0.790$) with Rao and Lohse's data [Heaps and Handel, 1999]. Our PTS constructed using co-occurrence has a better correlation with Rao and Lohse's space compared to that using Donderi's information measurement. This is expected because Rao and Lohse's space was developed using co-occurrence as well. We do not have Heaps and Handel's data for direct comparison. Nevertheless, Table 1 shows that the spaces are mutually consistent but not the same. The difference between the space of Rao and Lohse and our PTS can be attributed to the normalization of texture images. Thus, the PTS can be regarded as a reliable measurement of human's perception of texture similarity. In the following sections, the 4D PTS constructed based on information measurement will be used as the standard perceptual space because it has a better fit to the psychological data than that developed using co-occurrence.

4 Consistency of Computational Features

This section addresses the second critical issue: assessment of the perceptual consistency of computational texture features and similarity measurements. First, let us review commonly used texture features and similarity measurements.

4.1 Computational Texture Features

Statistical and spectral texture models are commonly used in existing image retrieval systems. Statistical models categorize textures according to statistical measurements of visual qualities such as coarseness and granularity. Spectral models characterize textures based on Fourier spectrum or filtering results. The statistical features proposed by Tamura et al. [Tamura *et al.*, 1978] are used in IBM's QBIC system. The Wold model [Liu and Picard, 1996] based on Fourier transform (spectral model) and Multiresolution Simultaneous Autoregressive model (MRSAR, statistical model) is used in MIT's PhotoBook. Gabor features (spectral model) are used in Ma and Manjunath's NeTra [Ma and Manjunath, 1996] and Leow and Lai's invariant texture space [Leow and Lai, 2000].

The above features are easy to compute, but the texture similarity computed based on these features are not necessarily consistent to human's perception. On the other hand, Santini and Jain developed the *fuzzy features contrast model* (FFC) [Santini and Jain, 1999] which has the potential of being perceptually consistent. Their similarity measurement is based on Tversky's feature contrast model [Tversky, 1977] which can account for various peculiarities of human's perceptual similarity. They have applied FFC to measure texture similarity based on Gabor features, and have obtained encouraging results. Unfortunately, due to combinatorial problem, they have compared FFC's measurements with only a small number of human ranking data, and are not able to perform more thorough comparison with human perception.

FFC defines a fuzzy membership function μ_{ki} for each Gabor feature G_{ki} of texture t_k by a sigmoid function:

$$\mu_{ki} = \left(1 + \exp\left(-\frac{G_{ki} - \bar{G}_i}{\gamma \sigma_i}\right)\right)^{-1} \tag{11}$$

where $\bar{G}_i$ is the mean feature of the textures along the i-th dimension, σ_i is the corresponding standard deviation, and γ is a constant parameter (which is fixed at 1 in [Santini and Jain, 1999]). The similarity between two Gabor features G_{ki} and G_{li} is then defined as

$$S(G_{ki}, G_{li}) = \sum_i \min\{\mu_{ki}, \mu_{li}\} - \alpha \sum_i \max\{\mu_{ki} - \mu_{li}, 0\} - \beta \sum_i \max\{\mu_{li} - \mu_{ki}, 0\} \tag{12}$$

where α and β are parameters. Various ranges of parameter values were tried. The values that produced the best match between FFC and PTS are $\alpha = \beta = 50, \gamma = 0.001$.

The major difficulty with using FFC is that the values of the parameters must be carefully selected to match human's perception. The authors acknowledge that the problem of determining the parameter values has not been adequately addressed [Santini and Jain, 1999].

4.2 Perceptual Consistency Tests

Perceptual consistency is assessed by comparing the texture distances measured by various features with that measured

Table 2: Comparison of various computational features and similarity measurements with PTS. Mahalanobis distance is used for MRSAR in the Wold model. r = Pearson's correlation coefficient, e = mean squared error.

feature	distance	r	e
Tamura	Euclidean	0.251	0.132
Gabor	Euclidean	0.273	0.131
Gabor	FFC	0.330	0.115
MRSAR	Euclidean	0.144	0.139
MRSAR	Mahalanobis	0.061	0.152
invariant	Euclidean	0.222	0.136

in the 4D PTS. Both Pearson's correlation coefficient (Eq. 5) and the mean squared error (MSE) of linear regression (Eq. 2) are used as the measurement of consistency. The larger the Pearson's coefficient, or the smaller the MSE, the more consistent computational features are to PTS (Section 2).

The following features are assessed: Tamura's features, Gabor, MRSAR, and invariant texture space. We could not assess the Wold model directly because it defines only a ranking order of the textures without explicit measurement of texture similarity. Therefore, only the MRSAR features used in the Wold model are assessed. The invariant texture space method maps Gabor features to a space that is invariant to texture scales and orientations [Leow and Lai, 2000], whereas the other features are not invariant. So, it is interesting to see how consistent is the invariant space to PTS.

Table 2 summarizes the results, and shows that Gabor feature and Gabor with FFC are most consistent with PTS. In particular, measuring Gabor similarity with FFC does improve Gabor feature's consistency. Measuring MRSAR similarity with Euclidean distance is perceptually more consistent than measuring with Mahalanobis distance. The degrees of consistency of computational features ($r \approx 0.3$) are, however, not very high compared to those between various perceptual spaces (Table 1, $r \approx 0.7$). Therefore, it can be concluded that these computational features and similarity measurements are not very consistent with human's perception.

5 Mapping Features to Perceptual Space

The previous section demonstrated that computational texture features are not very consistent with human's perception as measured in the 4D PTS. This section addresses the third critical issue: Assessment of the accuracy of mapping computational texture features to PTS.

The feature mapping problem is to find a set of functions f_k, $k = 1, \ldots, 4$, that map the computational features $\mathbf{x}$ of a texture into the coordinates y_k in PTS. That is, $y_k = f_k(\mathbf{x})$. Our study reveals that the functions f_k are typically nonlinear, and the form of the nonlinearity cannot be readily determined. Therefore, both multilayer neural networks and Support Vector Machine (SVM) regression were tested. SVM regression was found to give more accurate results. In addition, it has fewer parameters to tune than do neural networks. So, only the results of SVM regression are reported here.

SVM regression maps a multidimensional input to a single output value. Thus, four separate SVMs are required to map computational features $\mathbf{x}$ to the four coordinates $y_k, k = 1, \ldots, 4$, of PTS:

$$y_k = \sum_i (\alpha_{ki} - \alpha^*_{ki})\, g(\mathbf{v}_{ki}, \mathbf{x}) + b_k \qquad (13)$$

where α_{ki} and α^*_{ki} are the Lagrange multipliers (analogous to the weights of neural networks), g is a kernel function, $\mathbf{v}_{ki}$ are the support vectors, and b_k is the bias term. We experimented with various kernel functions and found that the Gaussian kernel consistently produced more accurate results. Various parameter values of the Gaussian kernel were tried, and it was found that different computational features require different parameter values for optimal mapping.

Table 3: Testing errors of feature mapping. Columns 2 to 5 are the average testing errors of n-fold cross-validation. 1D denotes the mean squared error (MSE) along one of the four dimensions of PTS. 4D denotes the overall MSE of the 4D coordinates in PTS. Columns 6 and 7 are the results of comparing the distances in PTS with those measured by the features *after* they are mapped to PTS.

	unknown instance		unknown texture		distance comparison	
features	1D	4D	1D	4D	r	e
Tamura	0.036	0.144	0.064	0.256	0.857	0.020
Gabor	0.0013	0.0052	0.054	0.216	0.862	0.018
MRSAR	0.024	0.096	0.061	0.244	0.859	0.019
invariant	0.031	0.124	0.067	0.268	0.632	0.031

The assessment of perceptual consistency used the 60 normalized texture images described in Section 3.3 as the data set. Each image was cropped into 9 subimages of size 128×128 pixels, resulting in a total of 540 texture samples. To better assess the performance of SVM, two different training and testing experiments were conducted:

1. Unknown instance of known texture.
   In this experiment, 8 of the 9 samples of each texture were used for training, and the remaining 1 sample was used for testing. Although the testing samples were different from the training samples, they were perceptually similar to some of the training samples. Therefore, the testing error was expected to be small. 9-fold cross-validation tests were conducted.
2. Unknown texture.
   This experiment tests SVM's performance on unknown texture. It is an important test because new textures can appear in real applications. In this experiment, all the samples of 54 (90%) of the 60 textures were used for training, and all the samples of the other 6 (10%) textures were used for testing. Since the testing samples were perceptually different from the training samples, the testing error was expected to be larger than that of the first experiment. 10-fold cross-validation tests were conducted.

Table 3 illustrates the mapping results obtained by SVM regression. As expected, for all the features, testing errors for unknown instance is smaller than those for unknown texture.

Moreover, being most consistent with PTS (Table 2), Gabor features can be mapped to PTS more accurately than other features can. In particular, Gabor features' testing error for unknown instance is much smaller than those of other features and those for unknown texture tests. Therefore, it can be concluded that accurate mapping to PTS can be achieved, at least for Gabor features and under the condition that some samples of all texture types appear in the training set.

After mapping computational features to PTS, one would expect the mapped coordinates to be more consistent with PTS. A comparison is performed between PTS and the computational features mapped by SVM models trained under the second condition (unknown texture). The distance correlation results are shown in Table 3. Comparing Table 3 with Table 2 shows that mapping computational features to PTS does improve the perceptual consistency of the features.

6 Application Examples

Given an accurate mapping, it is now possible to use PTS for various applications, such as, retrieval, classification, and classification with uncertainty.

1. In retrieval, we like to sort a set of textures into an ordered list $\{t_i\}$ such that, with respect to the query texture t_0, $p_{0i} \leq p_{0j}$ for any $i < j$, i.e., textures that are perceptually similar to t_0 are placed at the beginning of the list. This problem can be solved by mapping t_0 to PTS and ordering the textures t_i according to d_{0i}.
2. In classification, we like to classify a texture t_0 to a class C such that for any $t_i \in C$ and any $t_j \notin C$, $p_{0i} \leq p_{0j}$. This can be achieved by classifying t_0 to the class, say, C of its nearest neighbor, say, t_k, i.e., $d_{0k} \leq d_{0i}$ for any $t_i \neq t_k$. Since the distance between textures in the same class is smaller than the distance between textures in different classes, by perceptual consistency, $p_{0i} \leq p_{0j}$ for any $t_i \in C$ and any $t_j \notin C$.
3. The distance measurement d_{ij} in PTS can be used to represent uncertainty of classification. It is computed in terms of the independent information $H(t_i|t_j)$ and $H(t_j|t_i)$ (Eq. 8), which represents the uncertainty of classifying t_i given t_j and vice versa.

7 Conclusion

This paper has addressed three critical issues about perceptually consistent texture analysis:

1. A 4D perceptual texture space (PTS) has been developed. By comparing with existing perceptual spaces, it is verified that the PTS provides a reliable measurement of human's perception of texture similarity.
2. It is shown, by correlating with the distances measured in PTS, that commonly used computational texture features and similarity measurements are not very consistent with PTS. Among the various features, Gabor features and Gabor with FFC give the highest correlation.
3. It is shown that mapping computational features to PTS improves the features' perceptual consistency. In particular, Gabor features can be most accurately mapped to PTS, especially when some samples of all texture types appear in the training set.

This paper thus establishes that PTS can be used as a yardstick for assessing the perceptual consistency of computational features and similarity measurements.

For practical applications, it might be necessary to find a method that can map computational features to PTS accurately under the more stringent condition of unknown texture types. This would be necessary if some rare textures that are distinct from the known textures used in the training process appear during actual run. In addition, it would be necessary to perform the mapping in a manner that is invariant to texture scale and orientation.

References

[Amadasun and King, 1989] M. Amadasun and R. King. Textural features corresponding to textural properties. *IEEE Trans. SMC*, 19(5):1264–1274, 1989.

[Donderi, 1988] D. C. Donderi. Information measurement of distinctiveness and similarity. *Perception and Psychophysics*, 44(6):576–584, 1988.

[Hair *et al.*, 1998] Joseph. F. Hair, R. E. Anderson, and R. L. Tatham. *Multivariate Data Analysis*. Prentice Hall, 1998.

[Heaps and Handel, 1999] C. Heaps and S. Handel. Similarity and features of natural textures. *J. Expt. Psycho.: Human Perception and Performance*, 25(2):299–320, 1999.

[Leow and Lai, 2000] W. K. Leow and S. Y. Lai. Scale and orientation-invariant texture matching for image retrieval. In M. K. Pietikäinen, editor, *Texture Analysis in Machine Vision*. World Scientific, 2000.

[Liu and Picard, 1996] F. Liu and R.W. Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE Trans. PAMI*, 18(7):722–733, 1996.

[Ma and Manjunath, 1996] W. Y. Ma and B. S. Manjunath. Texture features and learning similarity. In *Proc. of IEEE Conf. CVPR*, pages 1160–1169, 1996.

[Meyer and Greenberg, 1987] G. W. Meyer and D. P. Greenberg. Perceptual color spaces for computer graphics. In H. J. Durett, editor, *Color and the Computer*. Academic Press, Lodon, 1987.

[Rao and Lohse., 1993] A. R. Rao and G. L. Lohse. Towards a texture naming system: Identifying relevant dimensions of texture. In *IEEE Conf. on Visualization*, pages 220–227, 1993.

[Santini and Jain, 1999] S. Santini and R. Jain. Similarity measures. *IEEE Trans. PAMI*, 21(9):871–883, 1999.

[Schiffman, 1996] H. R. Schiffman. *Sensation and Perception: An Integrated Approach*. John Wiley & Sons, 4 edition, 1996.

[Tamura *et al.*, 1978] H. Tamura, S. Mori, and T. Yamawaki. Textural features corresponding to visual perception. *IEEE Trans. SMC*, 8(6):460–47, 1978.

[Tversky, 1977] A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

Fuzzy Conceptual Graphs for Matching Images of Natural Scenes

Philippe Mulhem†, Wee Kheng Leow‡, and Yoong Keok Lee‡
† IPAL-CNRS, National University of Singapore
‡ School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543
mulhem, leowwk, leeyoong@comp.nus.edu.sg *

Abstract

Conceptual graphs are very useful for representing structured knowledge. However, existing formulations of fuzzy conceptual graphs are not suitable for matching images of natural scenes. This paper presents a new variation of fuzzy conceptual graphs that is more suited to image matching. This variant differentiates between a model graph that describes a known scene and an image graph which describes an input image. A new measurement is defined to measure how well a model graph matches an image graph. A fuzzy graph matching algorithm is developed based on error-tolerant subgraph isomorphism. Test results show that the matching algorithm gives very good results for matching images to predefined scene models

1 Introduction

Conceptual graphs [Sowa, 1984; 1990; 1993; 2000] are expressive for structured knowledge representation and they can be converted to first-order logic expressions. Hence, they are used with theoretical logic-based approaches in applications like hypertext modeling [Kherbeik and Chiaramella, 1995] and image representation and retrieval [Mechkour, 1995; Ounis and Pasça, 1998]. However, in these applications, uncertainty is not represented in the conceptual graphs.

To incorporate uncertainty, Morton [Morton, 1987] extended Sowa's conceptual graph theory to fuzzy conceptual graphs, which include fuzzy referents, fuzzy operations, and fuzzy open and closed worlds. Wuwongse and colleagues [Wuwongse and Manzano, 1993; Wuwongse and Tru, 1996] extended Morton's fuzzy conceptual graphs to incorporate fuzzy conceptual relations. In addition, Maher [Maher, 1991] proposed a similarity measurement for matching simple fuzzy conceptual graphs.

Existing formulations of conceptual graphs are important developments of the theory of fuzzy conceptual graphs. However, they are not suitable for matching uncertain information in images with model scenes. For example, in existing formulations, a concept has only one type. In practice, however, it is very difficult to accurately determine the concept types of the regions in an image. In [Petrakis and Faloutsos, 1997], Attributed Relational Graphs are used for image content representation, and [Medasani and Krishnapuram, 1999] proposed fuzzy matching of these graphs. But, both methods do not handle hierarchies of concept types and of relations.

This paper presents a new variation of fuzzy conceptual graphs that is more suited to matching images of natural scenes. This variant has several distinct characteristics that distinguish it from existing formulations:

- It associates a concept with multiple concept types so as to handle the uncertainty of classification.
- It introduces a new component called *relation attribute* for capturing concepts such as the ratio between the sizes of two image regions. Separating relation attributes from relations allows two or more correlated relations to share the same attribute explicitly. Relationships between attributes can be represented as well. For example, we can explicitly represent the fact that the ratio between the sizes of two objects is larger than the size ratio of two other objects.
- It differentiates between *model graphs* and *image graphs*. A model graph describes a model of a scene whereas an image graph describes the components and their relationships in an image. As will be described in more details in Section 2, these two types of fuzzy graphs have different characteristics.
- It defines a similarity measurement between a model graph and an image graph in terms of graph projection. In addition, an algorithm is implemented to compute the similarity based on subgraph isomorphism.

2 Fuzzy Conceptual Graphs

This section first introduces and defines the basic elements of a fuzzy conceptual graph. Next, the degree of match between conceptual graphs is defined and, finally, the graph matching algorithm is described.

2.1 Basic Elements

A *fuzzy conceptual graph* $G(C, R, A)$ is a directed graph formed from three kinds of components: *fuzzy concepts*, *fuzzy relations*, and *fuzzy relation attributes*. The set C is the set of fuzzy concepts, R the set of fuzzy relations, and A the set of fuzzy relation attributes.

*This research is supported by NUS ARF R-252-000-049-112 and joint NSTB and NUS ARF RP3999903.

A *fuzzy concept* c is a 3-tuple $[T, e, f]$ that consists of a set T of concept types t_i, a referent e, and a fuzzy membership function $f(t_i)$, $0 \leq f(t_i) \leq 1$ for each t_i. In [Wuwongse and Manzano, 1993; Wuwongse and Tru, 1996], $f(t_i)$ is called the *measure* assigned to the type t_i. In practice, $f(t_i)$ can be computed as the confidence of classifying a region as type t_i. A *crisp concept* c can be regarded as a special kind of fuzzy concept whose set T consists of only one concept type t with $f(t) = 1$. For notational convenience, we denote a crisp concept as a 3-tuple of the form $[t, e, 1]$.

A *fuzzy relation* r is a 2-tuple (t, v) that consists of a crisp relation type t and a fuzziness value v which represents the probability of occurrence of the relation r in the images. A *crisp relation* is a special kind of fuzzy relation of the form $(t, 1)$, i.e., $v = 1$ which means that the relation is not fuzzy.

A *fuzzy relation attribute* a is a 3-tuple $[t, e, f]$ that consists of a crisp relation attribute type t, a referent e, and a fuzzy membership function $f(e)$ such that $0 \leq f(e) \leq 1$. A *crisp relation attribute* is a special kind of fuzzy relation attribute of the form $[t, e, 1]$, where $f(e) = 1$.

The *denotation* $\mathcal{D}(t)$ of type t (concept, relation, or relation attribute type) is the set of all possible individual referents of t. A type t' is a *specialization* or *subtype* of type t, denoted $t' \leq t$, if $\mathcal{D}(t') \subseteq \mathcal{D}(t)$ [Sowa, 1984]. The reciprocity holds also. Note that the specialization relationship is reflexive and transitive but not symmetric. The set of concept types together with the specialization relationship forms a finite *lattice of concept types*. Relation attributes are special kinds of concepts. So, the set of relation attribute types form a sub-lattice in the lattice of concept types. In the same way, the set of relation types forms a finite lattice of relation types.

A *model graph* represents a model of a known scene. In a model, the concept types are known. For example, a mountain-and-lake scene contains sky, mountain, water, tree, etc. (Fig. 1) Therefore, the concepts in a model graph are crisp concepts. However, the relations between the concepts may vary from one image to the next. Moreover, to capture fuzzy linguistics such as "much smaller" and "slightly smaller", the relations and relation attributes have to be fuzzy. For example, the relation $(\text{comp}, *, 0.9)$ (represented as $(\text{comp} \mid 0.9)$ in Fig. 1) from [tree] to [image] indicates that trees appear in 90% of the images of mountain-and-lake scene. The relation $(\text{smaller}, *, 0.7)$ (represented as $(\text{smaller} \mid 0.7)$ in Fig. 1), with attribute $[\text{ratio}, *, \text{f0.5}]$ (represented as $[\text{ratio} \mid \text{f0.5}]$ in Fig. 1), from [tree] to [mountain] indicates that in 70% of the images of mountain-and-lake scene, the area occupied by trees is 50% that of the mountains. The symbol f0.5 denotes a fuzzy membership function that peaks at 0.5. If the ratio v in an input image is not exactly 0.5, then f5.0(v) will be smaller than the maximum value of 1.0.

Several relations can connect the same two concepts in a model graph. Among these relations, those that have the same relation type are regarded as forming a disjunctive group (i.e., OR) of relations (e.g., the "smaller" relation between tree and mountain in Fig. 1). The various disjunctive groups then form a conjunction (i.e., AND) of relations.

An *image graph* represents the structure of a particular input image. Due to the inherent uncertainty in the algorithms that classify the image regions, the concepts in an image graph are fuzzy. However, the relations between the regions in an image can be computed exactly. Therefore, the relations and their attributes are not fuzzy. For example, in Fig. 2, due to the ambiguity between leaves texture and grass texture, concept #e1 has a *confidence level* of 0.7 of being a leave texture or a tree texture, and a lower confidence level of 0.3 of being a grass texture. The rectangular box for #e1 represents the fuzzy concept $[T, \#e1, f]$ where $T = \{\text{leave, grass, tree}\}$ and f(leave) = 0.7, f(grass) = 0.3, and f(tree) = 0.7. The relation $(\text{comp} \mid 1)$ from concept #e1 to #e0 is crisp because the region #e1 is a component of the image #e0. Similarly, the relation $(\text{smaller} \mid 1)$ and the relation attribute $[\text{ratio} : 0.45 \mid 1]$ are crisp because the ratio of 0.45 is measured according to the areas of the regions in the image. In summary, a model graph contains crisp concepts, fuzzy relations, and fuzzy relation attributes, while an image graph contains fuzzy concepts, crisp relations, and crisp relation attributes (Table 1).

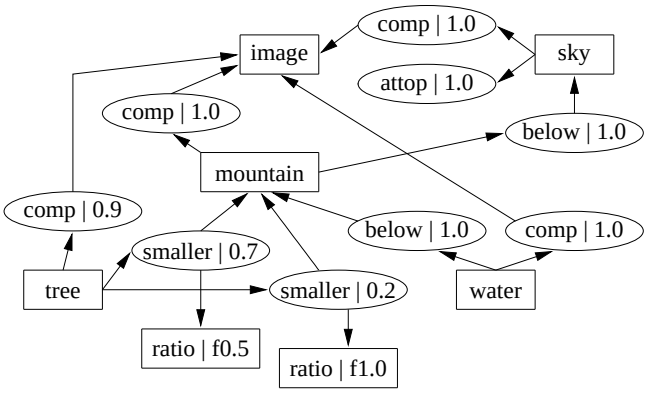


Figure 1: A (partial) model graph representing a mountain-and-lake scene. Relations are represented as ellipses. Concepts and relation attributes are represented as rectangles. Due to space limitation, only some of the concepts, relations, and relation attributes are shown.

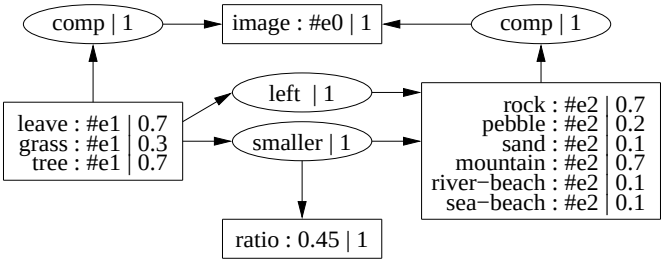


Figure 2: An example (partial) image graph of an input image. Relations are represented as ellipses. Concepts and relation attributes are represented as rectangles.

2.2 Fuzzy Graph Matching

The degree of match between a model graph and an image graph is defined in terms of *graph projection* [Sowa, 1984] and the degrees of match between their concepts, relations, and relation attributes. The degree of match M between a model concept $c = [t, *, 1]$ and an image concept

Table 1: Summary of the elements of fuzzy conceptual graphs for scene models and input images.

	element	notation	notation in figure
model graph	crisp concept	$[t, *, 1]$	$[t]$
	fuzzy relation	(t, v)	$(t \mid v)$
	fuzzy relation attribute	$[t, *, f]$	$[t \mid f]$
image graph	fuzzy concept	$[T, e, f]$	$[t_i : e \mid f(t_i)]$
	crisp relation	$(t, 1)$	$(t \mid 1)$
	crisp relation attribute	$[t, e, 1]$	$[t : e \mid 1]$

$c' = [T, e, f]$ is defined as

$$M(c, c') = \begin{cases} \max\limits_{t_i \le t} f(t_i) & \text{if } \exists\, t_i \in T \text{ such that } t_i \le t \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The degree of match M between a model relation $r = (t, v)$ and an image relation $r' = (t', 1)$ is defined as

$$M(r, r') = \begin{cases} v & \text{if } t' \le t \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

The degree of match M between a model relation attribute $a = [t, *, f]$ and an image relation attribute $a' = [t', e', 1]$ is defined as

$$M(a, a') = \begin{cases} f(e') & \text{if } t' \le t \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

A *projection* π from a fuzzy graph $G(C, R, A)$ to another fuzzy graph $G'(C', R', A')$ is a mapping such that

- for each concept c in C, $\pi(c)$ is a concept in C' and $M(c, \pi(c)) > 0$,
- for each relation r in R, $\pi(r)$ is a relation in R' and $M(r, \pi(r)) > 0$,
- for each relation attribute a in A, $\pi(a)$ is a relation attribute in A' and $M(a, \pi(a)) > 0$,
- for any two concepts or relation attributes s and t related by a relation r in G, $\pi(s)$ and $\pi(t)$ are related by $\pi(r)$ in G'.

In general, the projection $\pi(G)$ may not exist but the projection $\pi(g)$ from a subgraph g of G may exist. Therefore, the matching problem becomes one of finding the best matching subgraph g of G that maximizes $M_G(g, \pi(g))$.

Given a projection π from a subgraph $g(C, R, A)$ of model graph G to an image graph G', the degree of match M_G between g and $\pi(g)$ is defined as

$$M_G(g, \pi(g)) = \frac{1}{|G|} \left[\alpha_c \sum_{c_i \in C} M(c_i, \pi(c_i)) + \alpha_r \sum_{r_i \in R} M(r_i, \pi(r_i)) + \alpha_a \sum_{a_i \in A} M(a_i, \pi(a_i)) \right] \quad (4)$$

where $|G|$ is the sum of the number of concepts, the number of disjunctive groups of relations, and the sum of the maximum number of relation attributes in each disjunctive group of relation in the model graph G. The second summation in Eq. 4 sums over the conjunctive groups of relations. Note that if $g = G$ and $\alpha_c = \alpha_r = \alpha_a = 1$, then $M_G(g, \pi(g)) = M_G(G, \pi(G)) = |G|/|G| = 1$. That is, a perfect match yields a matching value of 1. We set $\alpha_c = \alpha_r = \alpha_a = 1$ in our test.

2.3 Graph Matching Algorithm

To facilitate the matching process, a graph is first decomposed into a set of *arches*, each consisting of a source concept s, a target concept t, a relation r from s to t, and zero or more relation attributes a_i. Thus, arches are actually subgraphs of the model graph or image graph. The match value between a model arch h and an image arch h' is simply the normalized match values of the concepts, relation, and relation attributes of the arches (i.e., Eq. 4).

Our fuzzy graph matching algorithm is a variant of Messmer and Bunke's error-tolerant subgraph isomorphism algorithm [Messmer and Bunke, 1998]. The main differences between these algorithms include the following:

- In [Messmer and Bunke, 1998], the smallest unit of operation is a vertex. In our algorithm, the smallest unit is an arch.
- For error-tolerant graph matching, our algorithm needs to consider only deletion of graph vertices. It does not require addition and replacement of vertices and edges, which are supported in [Messmer and Bunke, 1998].
- Two queues are maintained, one for normal search, and the other one for backtracking to the first-level subgraphs (just below the root). This method improves the quality of the solution found by the algorithm.

The fuzzy graph matching algorithm is as follows:

Convert model graph G to arch set H, image graph G' to arch set H'.
For each pair $(g, g') \in H \times H'$
  Compute $M_G(g, g')$.
  If $M_G(g, g') >$ threshold, then insert (g, g') into 1st-level queue Q_1 in decreasing order of $M_G(g, g')$.
Repeat n times
  Empty normal search queue Q_n.
  Remove first pair (g, g') from Q_1 and insert into Q_n.
  Repeat
    Remove first pair (g, g') from Q_n.
    If all arches in H have been considered for connecting to g, then break out of the inner repeat loop.
    For each pair $(h, h') \in H \times H'$ such that $M_G(h, h') > 0$, h can connect to g, and h' can connect to g'
      Connect h to g, h' to g'.
      Compute $M_G(g, g')$.
      If $M_G(g, g') >$ threshold, then insert (g, g') into Q_n in decreasing order of $M_G(g, g')$.
  Save (g, g') and the match value as a candidate solution.
Return the candidate solution with the highest match value.

The basic idea is to start off with a single arch in the model graph that best matches an arch in the image graph. This pair

of arches forms the initial matching model and image subgraphs. Subsequently, other model arches that have matching image arches are added to the matching subgraphs. For model arches that form a disjunctive group, only the best matching model arch is selected. This matching process repeats until all the arches in the model graph have been considered for addition to the matching subgraphs. Then, the matching model and image subgraphs are saved as a candidate solution.

Due to possible ambiguity of the concepts, initial erroneous matches that happen to have large match values may lead the search algorithm down the wrong search paths. So, another queue Q_1 is used to keep the first-level subgraphs. Once a candidate solution is found and saved, the normal search queue Q_n is emptied and a new search from another first-level subgraph is initiated. This method allows the algorithm to escape from local maxima of the solution space. This procedure is repeated n times, and the best solution among them is returned as the final matching result. In the tests reported in Section 3, n is set to 5.

The threshold in the algorithm is used to prune away subgraphs that have poor matching values. This method reduces the amount of searches that need to be considered and helps to improve the efficiency of the algorithm. In the tests, the threshold is set to 0.2.

During the execution of the algorithm, the value $M_G(g, g')$ can be computed incrementally. Let $p = g + h$ at c, i.e., connecting arch h to model graph g at the concept c results in the new model graph p. Note that $c \in g, h$. Similarly, let $p' = g' + h'$ at c'. Then, $M_G(p, p')$ can be computed incrementally as follows:

$$M_G(p, p') = M_G(g, g') + M_G(h, h') - \frac{1}{|G|} M(c, c') \ . \quad (5)$$

The time complexity of the matching algorithm is $O(n_a N_m^2 N_i (\log N_c + \log N_r))$ where N_m and N_i are the number of model and image arches respectively, n_a is the average number of relation attributes per relation, N_c is the number of concept types in the concept lattice, N_r is the number of relation types in the relation lattice. If each relation has at most one attribute (i.e. $n_a = 1$), then the time complexity reduces to $O(N_m^2 N_i (\log N_c + \log N_r))$.

3 Test Results

In our current implementation, an input image is segmented using a color and texture segmentation algorithm. The color histogram and Gabor texture features of each region are analyzed to classify the region into several possible classes, and the confidence level for each classification is measured. These measurements are used to generate the image graphs of images by a graph generation program. The model graphs of scene models are generated manually because, at present, we are still in the process of developing a learning algorithm for learning model graphs from example images.

Tests were conducted to match four scene models of different complexity (Fig. 3) with 200 images. A sample of the images are shown in Fig. 4. Table 2 summarizes the test results. Images that match the models well have higher match values than those that match the models poorly. Model a is

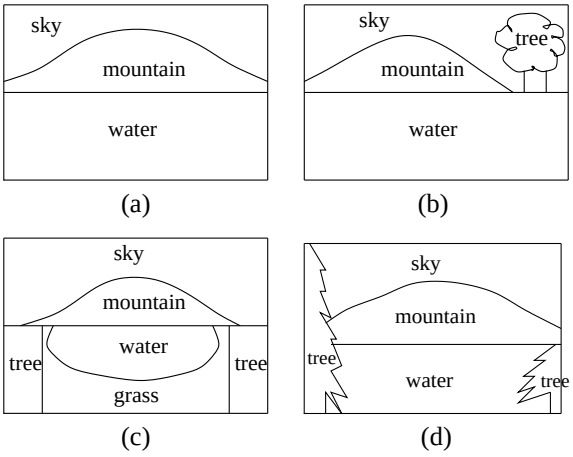


Figure 3: Schematic drawings of four scene models with different complexity.

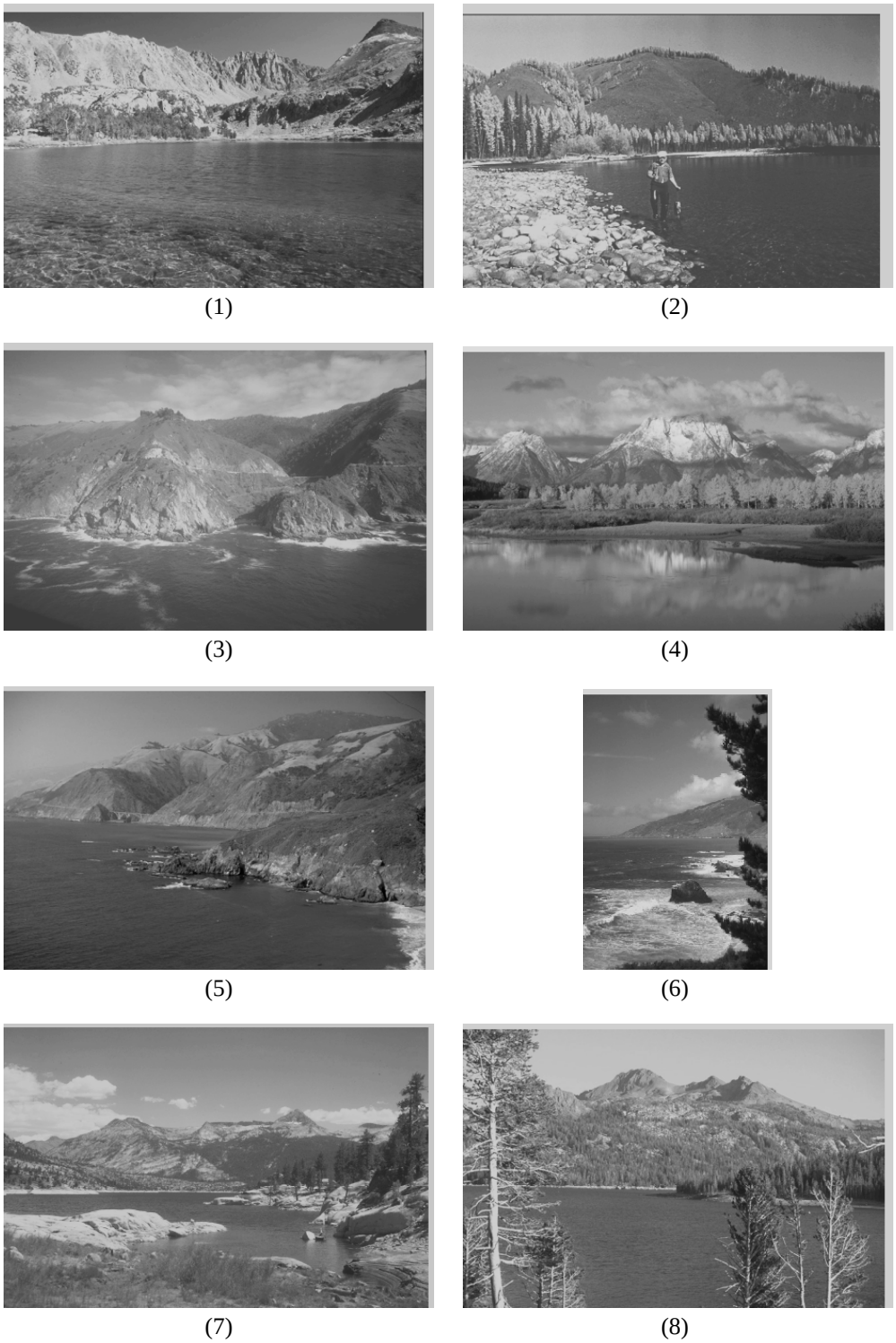
Table 2: Image matching results. Numbers in brackets are the number of concepts, relations, and relation attributes of the respective conceptual graphs. The other numbers are the match values.

image	model			
	a (4,12,0)	b (5,23,6)	c (7,41,12)	d (6,30,9)
1 (4,16,3)	0.838	0.482	0.191	0.321
2 (6,29,7)	0.814	0.671	0.343	0.471
3 (4,16,3)	0.814	0.444	0.177	0.367
4 (7,31,8)	0.795	0.482	0.290	0.421
5 (4,18,4)	0.767	0.444	0.181	0.387
6 (7,34,10)	0.686	0.666	0.354	0.485
7 (8,37,10)	0.623	0.621	0.432	0.541
8 (7,35,9)	0.471	0.465	0.403	0.895
9 (6,31,9)	0.519	0.371	0.313	0.638
10 (8,40,12)	0.448	0.409	0.337	0.405

the simplest among the four models. Most images can fully match model a successfully and thus have high match values. In particular, the first 5 images best match model a and have the highest match values. Model b differs from model a by having a small tree above the water. Therefore, it best matches images 2, 6, and 7.

Model c is the most complex model which has the most number of components. It is used to show that the algorithm can perform partial matching because none of the images can match model c fully. As a result, the images have smaller match values than those for matching the other models. Nevertheless, the ranking of the images is still consistent: images that match model c better have higher match values. In particular, images 7 and 8 match model c best.

Model d is used to illustrate an extreme case in which a very specific model is specified. In this test, model d is created according to image 8. The test results show that image 8 indeed matches model d very well. Image 9 differs a bit



(1) (2)

(3) (4)

(5) (6)

(7) (8)

Figure 4: Sample images used in the tests.

(9)

(10)

Figure 4: (Cont.) Sample images used in the tests.

from model d in that image 9 has tall trees on both sides of the image whereas model d has tall trees on only one side. Thus, image 9 has a lower match value than image 8. All other images cannot match model d very well and have very small match values.

4 Conclusion

This paper presented a new variant of fuzzy conceptual graphs that is more appropriate than existing formulations for representing and matching input images and model scenes. A new component called relation attribute is introduced into fuzzy conceptual graphs. Moreover, model graphs that describe known scenes are distinguished from image graphs that represent input images. A similarity measure is defined to assess the degree of match between model and image graphs. A graph matching algorithm is developed based on error-tolerant subgraph isomorphism algorithm. Experimental tests show that the matching algorithm gives very good results for matching images to scene models.

The fuzzy graphs can be input and modified manually using a graphical user interface. The image graphs of the images that match a model can be displayed by an image retrieval system for explaining how well the image matches the model. We are currently developing a method of learning model graphs given example images of the scenes.

References

[Kherbeik and Chiaramella, 1995] A. Kherbeik and Y. Chiaramella. Integrating hypermedia and information retrieval with conceptual graphs. In *Proc. Int. Conf. on Hypermedia, Information Retrieval and Multimedia*, pages 47–60, 1995.

[Maher, 1991] P. E. Maher. Conceptual graphs - a framework for uncertainty management. In *Proc. 10th Annual NAFIPS Conference*, pages 106–110, 1991.

[Mechkour, 1995] M. Mechkour. EMIR2: An extended model for image representation and retrieval. In *Proc. Int. Conf. on Databases and Expert Systems Applications (DEXA '95)*, pages 395–404, 1995.

[Medasani and Krishnapuram, 1999] S. Medasani and R. Krishnapuram. A fuzzy approach to content-based image retrieval. In *Proceedings of the IEEE Conference on Fuzzy Systems*, pages 1251–1257, Seoul, 1999.

[Messmer and Bunke, 1998] B. T. Messmer and H. Bunke. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Trans. PAMI*, 20(5):493–504, 1998.

[Morton, 1987] S. Morton. *Conceptual Graphs and Fuzziness in Artificial Intelligence*. PhD thesis, University of Bristol, 1987.

[Ounis and Pasça, 1998] I. Ounis and M. Pasça. RELIEF: Combinig evidences and rapidity into a signle system. In *Proc. ACM SIGIR '98*, pages 266–274, 1998.

[Petrakis and Faloutsos, 1997] G. M. Petrakis and C. Faloutsos. Similarity searching in large image databases. *IEEE Transactions on Knowledge and Data Engineering*, 9(3):435–447, 1997.

[Sowa, 1984] J. F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, 1984.

[Sowa, 1990] J. F. Sowa. Towards the expressive power of natural language. In *Proc. 5th Annual Workshop on Conceptual Graphs*, 1990.

[Sowa, 1993] J. Sowa. Relating diagrams to logic. In *Proc. Int. Conf. on Conceptual Structures (ICCS '93), (LNAI 699)*, pages 1–35, 1993.

[Sowa, 2000] J. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks/Cole Publisher, 2000.

[Wuwongse and Manzano, 1993] V. Wuwongse and M. Manzano. Fuzzy conceptual graph. In *Proc. Int. Conf. on Conceptual Structures (ICCS '93) (LNAI 699)*, pages 430–449, 1993.

[Wuwongse and Tru, 1996] V. Wuwongse and C. H. Tru. Towards fuzzy conceptual graph programs. In *Proc. Int. Conf. on Conceptual Structures (ICCS '96) (LNAI 1115)*, pages 263–276, 1996.

ROBOTICS AND PERCEPTION

PERCEPTION

Discriminating Animate from Inanimate Visual Stimuli

Brian Scassellati
MIT Artificial Intelligence Laboratory
200 Technology Square
Cambridge, MA 02139
scaz@ai.mit.edu

Abstract

From as early as 6 months of age, human children distinguish between motion patterns generated by animate objects from patterns generated by moving inanimate objects, even when the only stimulus that the child observes is a single point of light moving against a blank background. The mechanisms by which the animate/inanimate distinction are made are unknown, but have been shown to rely only upon the spatial and temporal properties of the movement. In this paper, I present both a multi-agent architecture that performs this classification as well as detailed comparisons of the individual agent contributions against human baselines.

1 Introduction

One of the most basic visual skills is the ability to distinguish animate from inanimate objects. We can easily distinguish between the movement of a clock pendulum that swings back and forth on the wall from the movement of a mouse running back and forth across the floor. Michotte [1962] first documented that adults have a natural tendency to describe the movement of animate objects in terms of intent and desire, while the movements of inanimate objects are described in terms of the physical forces that act upon them and the physical laws that govern them. Furthermore, by using only single moving points of light on a blank background, Michotte showed that these perceptions can be guided by even simple visual motion without any additional context.

Leslie [1982] proposed that this distinction between animate and inanimate objects reflects a fundamental difference in how we reason about the causal properties of objects. According to Leslie, people effortlessly classify stimuli into three different categories based on the types of causal explanations that can be applied to those objects, and different modules in the brain have evolved to deal with each of these types of causation. Inanimate objects are described in terms of mechanical agency, that is, they can be explained by the rules of *mechanics*, and are processed by a special-purpose reasoning engine called the *Theory of Body* module (ToBY) which encapsulates the organism's intuitive knowledge about how objects move. This knowledge may not match the actual physical laws that govern the movement of objects, but rather is our intuitive understanding of physics. Animate objects are described either by their actions or by their attitudes, and are processed by the *Theory of Mind* module which has sometimes been called an "intuitive psychology." System 1 of the theory of mind module (ToMM-1) explains events in terms of the intent and goals of agents, that is, their *actions*. For example, if you see me approaching a glass of water you might assume that I want the water because I am thirsty. System 2 of the theory of mind module (ToMM-2) explains events in terms of the *attitudes* and beliefs of agents. If you see me approaching a glass of kerosene and lifting it to my lips, you might guess that I believe that the kerosene is actually water. Leslie further proposed that this sensitivity to the spatio-temporal properties of events is innate, but more recent work from Cohen and Amsel [1998] may show that it develops extremely rapidly in the first few months and is fully developed by 6-7 months.

Although many researchers have attempted to document the time course of the emergence of this skill, little effort has gone into identifying the mechanisms of how an adult or an infant performs this classification. This paper investigates a number of simple visual strategies that attempt to perform the classification of animate from inanimate stimuli based only on spatio-temporal properties without additional context. These strategies have been implemented on a humanoid robot called Cog as part of an on-going effort to establish basic social skills and to provide mechanisms for social learning [Scassellati, 2000]. A set of basic visual feature detectors and a context-sensitive attention system (described in section 2) select a sequence of visual targets (see Figure 1). The visual targets in each frame are linked together temporally to form spatio-temporal trajectories (section 3). These trajectories are then processed by a multi-agent representation that mimics Leslie's ToBY module by attempting to describe trajectories in terms of naive physical laws (section 4). The results of the implemented system on real-world environments are introduced, and a comparison against human performance on describing identical data is discussed in section 5.

2 Visual Precursors

Cog's visual system has been designed to mimic aspects of an infant's visual system. Human infants show a preference for stimuli that exhibit certain low-level feature properties. For example, a four-month-old infant is more likely to look at a

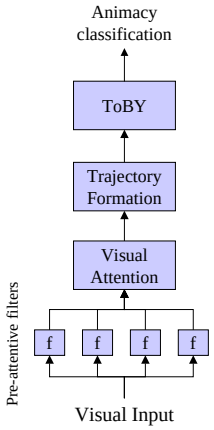


Figure 1: Overall architecture for distinguishing animate from inanimate stimuli. Visual input is processed by a set of simple feature detectors, each of which contributes to a visual attention process. Salient objects in each frame are linked together to form spatio-temporal trajectories, which are then classified by the "theory of body" (ToBY) module.

moving object than a static one, or a face-like object than one that has similar, but jumbled, features [Fagan, 1976]. Cog's perceptual system combines many low-level feature detectors that are ecologically relevant to an infant. Three of these features are used in this work: color saliency analysis, motion detection, and skin color detection. These low-level features are then filtered through an attentional mechanism which determines the most salient objects in each camera frame.

2.1 Pre-attentive visual routines

The color saturation filter is computed using an opponent-process model that identifies saturated areas of red, green, blue, and yellow [Itti *et al.*, 1998]. The color channels of the incoming video stream (r, g, and b) are normalized by the luminance l and transformed into four color-opponency channels (r', g', b', and y'):

$$r' = \frac{r}{l} - (\frac{g}{l} + \frac{b}{l})/2 \quad (1)$$

$$g' = \frac{g}{l} - (\frac{r}{l} + \frac{b}{l})/2 \quad (2)$$

$$b' = \frac{b}{l} - (\frac{r}{l} + \frac{g}{l})/2 \quad (3)$$

$$y' = \frac{\frac{r}{l} + \frac{g}{l}}{2} - \frac{b}{l} - \|\frac{r}{l} - \frac{g}{l}\| \quad (4)$$

The four opponent-color channels are thresholded and smoothed to produce the output color saliency feature map.

In parallel with the color saliency computations, The motion detection module uses temporal differencing and region growing to obtain bounding boxes of moving objects. The incoming image is converted to grayscale and placed into a ring of frame buffers. A raw motion map is computed by passing the absolute difference between consecutive images through a threshold function $\mathcal{T}$:

$$M_{raw} = \mathcal{T}(\|I_t - I_{t-1}\|) \quad (5)$$

This raw motion map is then smoothed to minimize point noise sources.

The third pre-attentive feature detector identifies regions that have color values that are within the range of skin tones [Breazeal *et al.*, 2000]. Incoming images are first filtered by a mask that identifies candidate areas as those that satisfy the following criteria on the red, green, and blue pixel components:

$$2g > r > 1.1g \qquad 2b > r > 0.9b \qquad 250 > r > 20 \quad (6)$$

The final weighting of each region is determined by a learned classification function that was trained on hand-classified image regions. The output is again median filtered with a small support area to minimize noise.

2.2 Visual attention

Low-level perceptual inputs are combined with high-level influences from motivations and habituation effects by the attention system. This system is based upon models of adult human visual search and attention [Wolfe, 1994], and has been reported previously [Breazeal and Scassellati, 1999]. The attention process constructs a linear combination of the input feature detectors and a time-decayed Gaussian field which represents habituation effects. High areas of activation in this composite generate a saccade to that location and compensatory neck movement. The weights of the feature detectors can be influenced by the motivational and emotional state of the robot to preferentially bias certain stimuli. For example, if the robot is searching for a playmate, the weight of the skin detector can be increased to cause the robot to show a preference for attending to faces. The output of the attention system is a labeled set of targets for each camera frame that indicate the positions (and feature properties) of the k most salient targets. For the experiments presented here, $k = 5$.

3 Computing Motion Trajectories

The attention system indicates the most salient objects at each time step, but does not give any indication of the temporal properties of those objects. Trajectories are formed using the multiple hypothesis tracking algorithm proposed by Reid [1979] and implemented by Cox and Hingorani [1996]. The centroids of the attention targets form a stream of target locations $\{P_t^1, P_t^2, ...P_t^k\}$ with a maximum of k targets present in each frame t. The objective is to produce a labeled trajectory which consists of a set of points, at most one from each frame, which identify a single object in the world as it moves through the field of view:

$$T = \{P_1^{i_1}, P_2^{i_2}, ...P_t^{i_n}\} \quad (7)$$

However, because the existence of a target from one frame to the next is uncertain, we must introduce a mechanism to compensate for objects that enter and leave the field of view and to compensate for irregularities in the earlier processing modules. To address these problems, we introduce phantom points that have undefined locations within the image plane but which can be used to complete trajectories for objects that enter, exit, or are occluded within the visual field. As each new point is introduced, a set of hypotheses linking that point

Figure 2: The last frame of a 30 frame sequence with five trajectories identified. Four nearly stationary trajectories were found (one on the person's head, one on the person's hand, one on the couch in the background, and one on the door in the background). The final trajectory resulted from the chair being pushed across the floor.

to prior trajectories are generated. These hypotheses include representations for false alarms, non-detection events, extensions of prior trajectories, and beginnings of new trajectories. The set of all hypotheses is pruned at each time step based on statistical models of the system noise levels and based on the similarity between detected targets. This similarity measurement is based on similarities of object features such as color content, size, and visual moments. At any point, the system maintains a small set of overlapping hypotheses so that future data may be used to disambiguate the scene. Of course, at any time step, the system can also produce the set of non-overlapping hypotheses that are statistically most likely. Figure 2 shows the last frame of a 30 frame sequence in which a chair was pushed across the floor and the five trajectories that were located.

4 The Theory of Body Module

To implement the variety of naive physical laws encompassed by the Theory of Body module, a simple agent-based approach was chosen. Each agent represents knowledge of a single theory about the behavior of inanimate physical objects. For every trajectory t, each agent a computes both an animacy vote α_{ta} and a certainty ρ_{ta}. The animacy votes range from $+1$ (indicating animacy) to -1 (indicating inanimacy), and the certainties range from 1 to 0. For these initial tests, five agents were constructed: an insufficient data agent, a static object agent, a straight line agent, an acceleration sign change agent, and an energy agent. These agents were chosen to handle simple, common motion trajectories observed in natural environments, and do not represent a complete set. Most notably missing is an agent to represent collisions, both elastic and inelastic.

At each time step, all current trajectories receive a current animacy vote V_t. Three different voting algorithms were tested to produce the final vote V_t for each trajectory t. The first voting method was a simple winner-take-all vote in which the winner was declared to be the agent with the greatest absolute value of the product: $V_t = \max_a \|\alpha_{ta} \times \rho_{ta}\|$ The second method was an average of all of the individual vote products: $V_t = \frac{1}{A}\sum_a(\alpha_{ta} \times \rho_{ta})$ where A is the number of agents voting. The third method was a weighted average of the products of the certainties and the animacy votes: $V_t = \frac{1}{A}\sum_a(w_a \times \alpha_{ta} \times \rho_{ta})$ where w_a is the weight for agent a. Weights were empirically chosen to maximize performance under normal, multi-object conditions in natural environments and were kept constant through out this experiment as 1.0 for all agents except the static object agent which had a weight of 2.0. The animacy vote at each time step is averaged with a time-decaying weight function to produce a sustained animacy measurement.

4.1 Insufficient Data Agent

The purpose of the insufficient data agent is to quickly eliminate trajectories that contain too few data points to properly compute statistical information against the noise background. Any trajectory with fewer than one-twentieth the maximum trajectory length or fewer than three data points is given an animacy vote $\alpha = 0.0$ with a certainty value of 1.0. In practice, maximum trajectory lengths of 60-120 were used (corresponding to trajectories spanning 2-4 seconds), so any trajectory of fewer than 3-6 data points was rejected.

4.2 Static Object Agent

Because the attention system still generates target points for objects that are stationary, there must be an agent that can classify objects that are not moving as inanimate. The static object agent rejects any trajectory that has an accumulated translation below a threshold value as inanimate. The certainty of the measurement is inversely proportional to the translated distance and is proportional to the length of the trajectory.

4.3 Straight Line Agent

The straight line agent looks for constant, sustained velocities. This agent computes the deviations of the velocity profile from the average velocity vector. If the sum of these deviations fall below a threshold, as would result from a straight linear movement, then the agent casts a vote for inanimacy. Below this threshold, the certainty is inversely proportional to the sum of the deviations. If the sum of the deviations is above a secondary threshold, indicating a trajectory with high curvature or multiple curvature changes, then the agent casts a vote for animacy. Above this threshold, the certainty is proportional to the sum of the deviations.

4.4 Acceleration Sign Change Agent

One proposal for finding animacy is to look for changes in the sign of the acceleration. According to this proposal, anything that can alter the direction of its acceleration must be operating under its own power (excluding contact with other objects). The acceleration sign change agent looks for zero-crossings in the acceleration profile of a trajectory. Anything with more than one zero-crossing is given an animacy vote with a certainty proportional to the number of zero crossings.

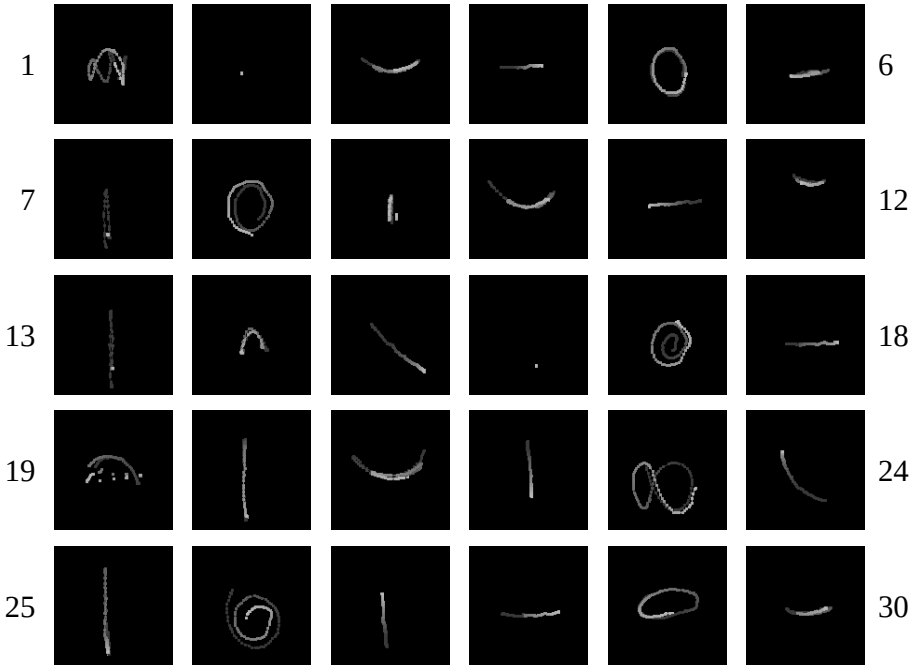


Figure 3: Thirty stimuli used in the evaluation of ToBY. Stimuli were collected by recording the position of the most salient object detected by the attention system when the robot observed natural scenes similar to the one shown in Figure 2. Each image shown here is the collapsed sequence of video frames, with more recent points being brighter than older points. Human subjects saw only a single bright point in each frame of the video sequence.

4.5 Energy Agent

Bingham, Schmidt, and Rosenblum [1995] have proposed that human adults judge animacy based on models of potential and kinetic energy. To explore their hypothesis, a simple energy model agent was implemented. The energy model agent judges an object that gains energy to be animate. The energy model computes the total energy of the system E based on a simple model of kinetic and potential energies:

$$E = \frac{1}{2}mv_y^2 + mgy \tag{8}$$

where m is the mass of the object, v_y the vertical velocity, g the gravity constant, and y the vertical position in the image. In practice, since the mass is a constant scale factor, it is not included in the calculations. This simple model assumes that an object higher in the image is further from the ground, and thus has more potential energy. The vertical distance and velocity are measured using the gravity vector from a three-axis inertial system as a guideline, allowing the robot to determine "up" even when its head is tilted. The certainty of the vote is proportional to the measured changes in energy.

5 Comparing ToBY's Performance to Human Performance

The performance of the individual agents was evaluated both on dynamic, real-world scenes at interactive rates and on more carefully controlled recorded video sequences.

For interactive video tasks, at each time step five attention targets were produced. Trajectories were allowed to grow to a length of sixty frames, but additional information on the long-term animacy scores for continuous trajectories were maintained as described in section 4. All three voting methods were tested. The winner-take-all and the weighted average voting methods produced extremely similar results, and eventually the winner-take-all strategy was employed for simplicity. The parameters of the ToBY module were tuned to match human judgments on long sequences of simple data structures (such as were produced by static objects or people moving back and forth throughout the room).

5.1 Motion Trajectory Stimuli

To further evaluate the individual ToBY agents on controlled data sequences, video from the robot's cameras were recorded and processed by the attention system to produce only a single salient object in each frame.[1] To remove all potential contextual cues, a new video sequence was created containing only a single moving dot representing the path taken by that

[1] This restriction on the number of targets was imposed following pilot experiments using multiple targets. Human subjects found the multiple target displays more difficult to observe and comprehend. Because each agent currently treats each trajectory independently, this restriction should not bias the comparison.

object set against a black background, which in essence is the only data available to the ToBY system. Thirty video segments of approximately 120 frames each were collected (see Figure 3). These trajectories included static objects (e.g. #2), swinging pendula (e.g. #3), objects that were thrown into the air (e.g. #7), as well as more complicated trajectories (e.g. #1). Figure 4 shows the trajectories grouped according to the category of movement, and can be matched to Figure 3 using the stimulus number in the second column. The third column of figure 4 shows whether or not the stimulus was animate or inanimate.

5.2 Human Animacy Judgments

Thirty-two adult, volunteer subjects were recruited for this study. Subjects ranged in age from 18 to 50, and included 14 women and 18 men. Subjects participated in a web-based questionnaire and were informed that they would be seeing video sequences containing only a single moving dot, and that this dot represented the movement of a real object. They were asked to rank each of the thirty trajectories shown in figure 3 on a scale of 1 (animate) to 10 (inanimate). Following initial pilot subjects (not included in this data), subjects were reminded that inanimate objects might still move (such as a boulder rolling down a hill) but should still be treated as inanimate. Subjects were allowed to review each video sequence as often as they liked, and no time limit was used.

The task facing subjects was inherently under-constrained, and the animacy judgments showed high variance (a typical variance for a single stimulus across all subjects was 2.15). Subjects tended to find multiple interpretations for a single stimulus, and there was never a case when all subjects agreed on the animacy/inanimacy of a trajectory. To simplify the analysis, and to remove some of the inter-subject variability, each response was re-coded from the 1-10 scale to a single animate (1-5) or inanimate (6-10) judgment. Subjects made an average of approximately 8 decisions that disagreed with the ground truth values. This overall performance measurement of 73% correct implies that the task is difficult, but not impossible. Column 4 of figure 4 shows the percentage of subjects who considered each stimulus to be animate. In two cases (stimuli #13 and #9), the majority of human subjects disagreed with the ground truth values. Stimulus #9 showed a dot moving alternately up and down, repeating a cycle approximately every 300 msec. Subjects reported seeing this movement as "too regular to be animate." Stimulus #13 may have been confusing to subjects in that it contained an inanimate trajectory (a ball being thrown and falling) that was obviously caused by an animate (but unseen) force.

5.3 ToBY Animacy Judgments

The identical video sequences shown to the human subjects were processed by the trajectory formation system and the ToBY system. Trajectory lengths were allowed to grow to 120 frames to take advantage of all of the information available in each short video clip. A winner-take-all selection method was imposed on the ToBY agents to simplify the reporting of the results, but subsequent processing with both other voting methods produced identical results. The final animacy judgment was determined to by the winning agent on the final time step. Columns 6 and 5 of figure 4 show the winning agent and that agent's animacy vote respectively.

Overall, ToBY agreed with the ground truth values on 23 of the 30 stimuli, and with the majority of human subjects on 21 of the 30 stimuli. On the static object categories, the circular movement stimuli, and the straight line movement stimuli, ToBY matched the ground truth values perfectly. This system also completely failed on all stimuli that had natural pendulum-like movements. While our original predictions indicated that the energy agent should be capable of dealing with this class of stimuli, human subjects seemed to be responding more to the repetitive nature of the stimulus rather than the transfer between kinetic and potential energy. ToBY also failed on one of the thrown objects (stimulus #20), which paused when it reached its apex, and on one other object (stimulus #19) which had a failure in the trajectory construction phase.

6 Conclusion

The distinction between animate and inanimate is a fundamental classification that humans as young as 6 months readily perform. Based on observations that humans can perform these judgments based purely on spatio-temporal signatures, this paper presented an implementation of a few simple naive rules for identifying animate objects. Using only the impoverished stimuli from the attentional system, and without any additional context, adults were quite capable of classifying animate and inanimate stimuli. While the set of agents explored in this paper is certainly insufficient to capture all classes of stimuli, as the pendulum example illustrates, these five simple rules are sufficient to explain a relatively broad class of motion profiles. These simple algorithms (like the agents presented here) may provide a quick first step, but do not begin to make the same kinds of contextual judgments that humans use.

In the future, we intend on extending this analysis to include comparisons against human performance for multi-target stimuli and for more complex object interactions including elastic and inelastic collisions.

Acknowledgments

Portions of this research were funded by DARPA/ITO under contract number DABT 63-99-1-0012, "Natural tasking of robots based on human interaction cues."

References

[Bingham *et al.*, 1995] Geoffrey P. Bingham, Richard C. Schmidt, and Lawrence D. Rosenblum. Dynamics and the orientation of kinematic forms in visual event recognition. *Journal of Experimental Psychology: Human Perception and Performance*, 21(6):1473–1493, 1995.

[Breazeal and Scassellati, 1999] Cynthia Breazeal and Brian Scassellati. A context-dependent attention system for a social robot. In *1999 International Joint Conference on Artificial Intelligence*, 1999.

Stimulus Category	Stimulus Number	Ground Truth	Human Judgment	ToBY Judgment	ToBY Expert	Notes
Static Objects	2	Inanimate	3%	Inanimate	Static Object	
	16	Inanimate	6%	Inanimate	Static Object	
Thrown Objects	7	Inanimate	44%	Inanimate	Energy	
	13	Inanimate	**53%**	Inanimate	Energy	
	20	Animate	78%	**Inanimate**	Straight Line	Pause at apex
	25	Animate	81%	Animate	Energy	Velocity increases near apex
Circular Movements	5	Animate	59%	Animate	Energy	
	8	Animate	81%	Animate	Energy	
	17	Animate	81%	Animate	Straight Line	
	26	Animate	78%	Animate	Acc. Sign Change	
	29	Animate	56%	Animate	Energy	
Straight Line Movements	4	Inanimate	47%	Inanimate	Straight Line	
	11	Inanimate	36%	Inanimate	Straight Line	
	22	Inanimate	30%	Inanimate	Straight Line	
	27	Animate	53%	Animate	Energy	moving up
	15	Inanimate	37%	Inanimate	Straight Line	
	24	Animate	75%	Animate	Energy	moving up and left
Pendula	3	Inanimate	16%	**Animate**	Energy	
	10	Inanimate	12%	**Animate**	Acc. Sign Change	
	21	Inanimate	31%	**Animate**	Acc. Sign Change	
	30	Inanimate	19%	**Animate**	Acc. Sign Change	
	12	Inanimate	6%	**Animate**	Acc. Sign Change	
Erratic Movements	1	Animate	97%	Animate	Energy	random movements
	6	Animate	75%	Animate	Acc. Sign Change	left/right bouncing
	9	Animate	**31%**	Animate	Acc. Sign Change	up/down bouncing
	14	Animate	75%	Animate	Acc. Sign Change	repeated left/right hops
	18	Animate	87%	Animate	Straight Line	delay at center point
	19	Animate	93%	**Inanimate**	Little Data	failure to track
	23	Animate	81%	Animate	Energy	figure-8
	28	Animate	90%	Animate	Straight Line	delay at center point

Figure 4: Comparison of human animacy judgments with judgments produced by ToBY for each of the stimuli from figure 3. Column 3 is the ground truth, that is, whether the trajectory actually came from an animate or inanimate source. Column 4 shows the percentage of human subjects who considered the stimulus to be animate. Column 5 shows the animacy judgment of ToBY, and column 6 shows the agent that contributed that decision. Bold items in the human or ToBY judgment columns indicate a disagreement with the ground truth.

[Breazeal *et al.*, 2000] Cynthia Breazeal, Aaron Edsinger, Paul Fitzpatrick, Brian Scassellati, and Paulina Varchavskaia. Social constraints on animate vision. *IEEE Intelligent Systems*, July/August 2000. To appear.

[Cohen and Amsel, 1998] Leslie B. Cohen and Geoffrey Amsel. Precursors to infants' perception of the causality of a simple event. *Infant Behavior and Develoment*, 21(4):713–732, 1998.

[Cox and Hingorani, 1996] Ingemar J. Cox and Sunita L. Hingorani. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(2):138–150, February 1996.

[Fagan, 1976] J. F. Fagan. Infants' recognition of invariant features of faces. *Child Development*, 47:627–638, 1976.

[Itti *et al.*, 1998] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 20(11):1254–1259, 1998.

[Leslie, 1982] Alan M. Leslie. The perception of causality in infants. *Perception*, 11:173–186, 1982.

[Michotte, 1962] A. Michotte. *The perception of causality*. Methuen, Andover, MA, 1962.

[Reid, 1979] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automated Control*, AC-24(6):843–854, December 1979.

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

[Wolfe, 1994] Jeremy M. Wolfe. Guided search 2.0: A revised model of visual search. *Psychonomic Bulletin & Review*, 1(2):202–238, 1994.

An Hybrid Approach to Solve the Global Localization Problem For Indoor Mobile Robots Considering Sensor's Perceptual Limitations

Leonardo Romero, Eduardo Morales and Enrique Sucar

ITESM, Campus Morelos, Temixco, Morelos, 62589, Mexico
lromero@zeus.umich.mx, {emorales,esucar}@campus.mor.itesm.mx

Abstract

Global localization is the problem of determining the position of a robot under global uncertainty. This problem can be divided in two phases: 1) from the sensor data (or *sensor view*), determine the set of locations where the robot can be; and 2) devise a strategy by which the robot can correctly eliminate all but the right location. The approach proposed in this paper is based on Markov localization. It applies the principal component method to get rotation invariant features for each location of the map, a Bayesian classification system to cluster the features, and polar correlations between the *sensor view* and the *local map views* to determine the locations where the robot can be. In order to solve efficiently the localization problem, as well as to consider the perceptual limitation of the sensors, the possible locations of the robot are restricted to be in a roadmap that keep the robot close to obstacles, and correlations between the possible local map views are pre–computed. The hypotheses are clustered and a greedy search determine the robot movements to reduce the number of clusters of hypotheses. This approach is tested using a simulated and a real mobile robot with promising results.

1 Introduction

Numerous tasks for a mobile robot require it to have a map of its environment and knowledge of where it is located in the map. Determining the location (position and orientation) of the robot in the environment is known as the *robot localization problem*. A recent survey [Borenstein *et al.*, 1996] illustrates the importance of the localization problem and illustrates the large number of existing approaches.

Traditionally, localization addresses two subproblems: *position tracking* and *global localization* [Thrun *et al.*, 1998]. Position tracking refers to the problem of estimating the location of the robot while it is moving. Drift and slippage impose limits on the ability to estimate the location of the robot within its global map. The robot knows its initial location and tracks its location after small movements using sensor data. Global localization is the problem of determining the location of the robot under global uncertainty. This problem arises, for example, when a robot uses a map that has been generated in a previous run, and it is not informed about its initial location within the map.

The global localization problem can be seen as consisting of two phases: hypothesis generation and hypothesis elimination [Dudek *et al.*, 1998]. The first phase is to determine the set of *hypothetical locations* H that are consistent with the sensing data obtained by the robot at its initial location. The second phase is to determine, in the case that H contains two or more hypotheses, which one is the true location of the robot, eliminating the incorrect hypotheses. Ideally, the robot should travel the minimum distance necessary to determine its exact location.

This paper presents an approach to solve the global localization problem in a known indoor environment modeled by an occupancy grid map, a two dimensional map where the environment is divided in square regions or cells of the same size. This approach is based on *Markov localization* (see [Gutmann *et al.*, 1998]) and takes into account the perceptual limitations of sensors in two ways: 1) the robot tries to keep a fixed distance to obstacles while the robot is moving and 2) the sensors have a limited range of operation. Following the approach described in [Romero *et al.*, 2000] to build a *roadmap* using the first restriction, the number of possible locations where the robot can be is significantly reduced. Taking into account the second restriction, *local map views* are computed from the map for each cell of the roadmap. The general form of each local map view can be compressed into two real numbers using the *principal component* method [Gonzalez and Wintz, 1987] with the advantage that these features are rotation independent. These compressed views are then clustered by Autoclass [Cheeseman and Stutz, 1996], a Bayesian classification system. The idea to generate the hypotheses is to compute the principal components of the sensor data (or sensor view), then use Autoclass to predict the best class (or classes) and finally perform polar correlations between the sensor view and the predicted local map views. To eliminate hypotheses, a greedy search is performed upon the correlations of the local map views of the same class.

The rest of this paper is organized as follows. Section 2 reviews some related relevant literature. Section 3 describes our approach to generate hypotheses. Section 4 presents the framework of Markov localization. Section 5 explains the approach to eliminate hypotheses. Experimental results using

a mobile robot simulator and a mobile robot with a low cost laser range sensor (implemented by a laser line generator and a camera) are shown in Section 6. Finally, some conclusions are given in Section 7.

2 Related work

Theoretical work on the subject has been reported in the literature. A solution to the hypothesis generation phase is given in [Guibas *et al.*, 1995]. In [Dudek *et al.*, 1998] it is shown that the problem of localizing a robot with minimum travel is NP–hard. These works make strong assumptions: first, perfect knowledge about the orientation of the mobile robot is considered; second, they assume the availability of noise–free sensor data.

Practical work has also been reported (see [Thrun, 1998; Castellanos and Tardos, 1999]). *Landmark methods* rely on the recognition of landmarks to keep the robot localized geometrically. While landmark methods can achieve impressive geometric localization, they require efficient recognition of features to use as landmarks. In contrast, *dense sensor methods* attempt to use whatever sensor information is available to update the location of the robot. They do this by matching dense sensor scans against a surface map of the environment, without extracting landmark features. See [Gutmann *et al.*, 1998] for an experimental comparison of two dense sensor matching methods: Kalman Filter and Markov localization. Some correlation techniques of laser range–finder scans are given in [Weib *et al.*, 1994]. In [Fox and Burgard, 1998] a method to move the robot in order to eliminate hypotheses is given. The idea is to move the robot so as to minimize future expected uncertainty, measured by the entropy of future belief distributions.

3 Hypothesis generation

In this section and the two following sections a simple occupancy grid map is used as an example to show the ideas behind the proposed approach. Figure 1 (a) shows this simple map built using a mobile robot simulator. Figure 1 (b) shows a local map view, extracted from the map for the position of the robot shown in (a), considering that the robot direction is aligned with a global fixed direction (pointing downwards in this case). Figure 1 (c) shows the sensor view (generated by the simulator) considering that the robot is aligned with the global direction. Both views have an angular resolution of 0.5 degrees, a common value found in laser range sensors, and include the robot position for reference as a black cell. Perceptual limitations are taken into account setting a maximum range of 3 meters.

The problem consist in estimating the set H of possible locations that have local map views consistent with the sensor view.

3.1 Polar correlation

To have a simple model of robot motion and hence a small state space in the Markov localization, we assume that the robot should be in one of 8 possible directions ($\theta_i = 45 * i$ degrees, $i = 0, ..7$), with respect to the global fixed direction, one for each adjacent cell. A polar correlation, using a sum of absolute differences, can be used to find the match between a local map view and the sensor view. Figure 2 shows the correlation results for all the possible angular displacements of the sensor view against the local map view shon in Figure 1. From the minimum difference an angular displacement can be computed to align the robot with one of the directions θ_i. Obviously, the right angular displacement should be indicated considering the most probable position of the robot. In the case of Figure 2 the angular displacement corresponds to -21 degres, and the best estimated direction is 270 degrees.

Figure 1: A simple environment. From left to right: (a) Occupancy grid map. (b) Local map view for the robot location showed in (a). (c) Sensor view

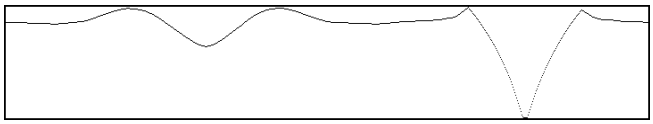


Figure 2: Correlation results. Angular displacements are from 0 (left) to 359.5 degrees (right).

As the Markov localization needs a probabilistic value $p(s|l)$ of perceiving a sensor view s given that the robot is at location l (cell $< x, y >$ with direction θ_i) a difference value $d(x, y, \theta_i)$ can be computed for direction θ_i from the correlation results between the sensor view s and the local map view at the cell $< x, y >$, and then a probabilistic value can be obtained from $d(x, y, \theta_i)$. We compute $d(x, y, \theta_i)$ as the minimum difference (in the correlation results) for an angular interval with center at θ_i. The desired probability is computed by $p(s|l) = e^{-\alpha d(x,y,\theta_i)}$ where α is positive real number.

Considering that this procedure is expensive, next sections show a fast procedure to find a small set of candidate cells to apply this procedure, instead of all the free cells in the map.

3.2 Roadmap

Following the ideas described in [Romero *et al.*, 2000], the set of possible cells, where the robot is allowed to move, tries to keep a fixed distance k to obstacles. It associates higher costs to free cells close to obstacles and the lowest costs to cells at distance k from obstacles. Figure 3 (a) shows the full set of free cells where the robot can be as white pixels, while Figure 3 (b) shows the cells that form the roadmap. There is a significant reduction in the number of cells. Let R be the set of cells of the roadmap.

In the next section we are going to consider only the cells $< x, y > \in R$.

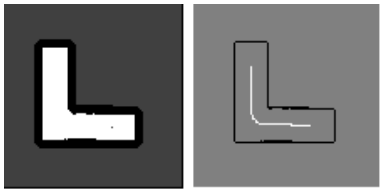


Figure 3: A roadmap. From left to right: (a) Full set of free cells. (b) The roadmap for k = 1 m.

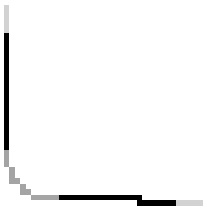


Figure 4: Clusters found by Autoclass. Different gray level are associated to different classes

3.3 Principal component analysis

We apply the principal component analysis described in [Gonzalez and Wintz, 1987] to find two rotation invariant features from a local map view or sensor view. The coordinates of each occupied cell of the view can be interpreted as two dimensional random variables with mean $\mathbf{m}$ and covariance $\mathbf{C}$. It is noted that $\mathbf{m}$ is a two dimensional vector and $\mathbf{C}$ is a 2×2 matrix. The eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ of $\mathbf{C}$ point in the directions of maximum variance (subject to the constraint that they are orthogonal) and the eigenvalues λ_1 and λ_2 are equal to the variance along eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ respectively. Our approach uses the two eigenvalues as rotation invariant features for a local map view or sensor view.

3.4 Bayesian clustering

Since the eigenvalues associated to adjacent cells of the roadmap normally do not change a lot, it makes sense to cluster cells according to their eigenvalues. For this purpose we use Autoclass [Cheeseman and Stutz, 1996]. Figure 4 show with different gray level the three classes found for the cells of the roadmap shown in Figure 3 (b). Autoclass found the same class for cells close to the end cells of the roadmap, another class for cells in the middle of the roadmap and the rest of cells in a third class.

At the beginning of the localization process, to take the robot to the roadmap, we apply the algorithm described in [Romero *et al.*, 2000] to learn a map of the environment. Using that approach it is easy to know if the robot is in the roadmap or need to execute some movements. Joining all these parts, the hypothesis generation does the following steps:

1. Compute the eigenvalues $\mathbf{e}$ from the sensor view.
2. With $\mathbf{e}$, use Autoclass to predict the most probable classes.

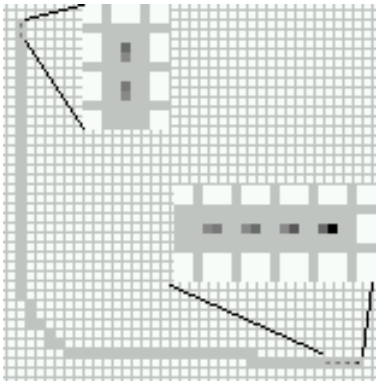


Figure 5: Most probable locations for the robot

3. For all cells in the classes found in the previous step, apply the correlation technique to get the angular displacement that align the robot with one of the possible directions and compute the conditional probabilities $p(s|l)$. The conditional probability for other cells of the roadmap are set to a very small value.

For the sensor view shown in Figure 1 (c) the conditional probabilities are shown in Figure 5. If the probability of finding the robot at location l of the roadmap is equal for all the cells in the roadmap, the same figure shows the most probable locations for the robot. The locations in the left top part of the figure point downwards and the locations in right bottom part of the figure point to the left. The following section describes the process to update the probability of hypotheses after the robot senses or moves.

4 Markov Localization

The key idea of Markov localization is to compute a probability distribution over all possible locations in the environment. $p(L_t = l)$ denotes the probability of finding the robot at location l at time t. Here, l is a location in $x - y - \theta_i$ space where x and y are Cartesian coordinates of cells and θ_i is a valid orientation. $p(L_0)$ reflects the initial state of knowledge and it is uniformly distributed to reflect the global uncertainty. $p(L_t)$ is updated whenever ...

1. ... *the robot moves.* Robot motion is modeled by a conditional probability, denoted by $p_a(l|l')$. $p_a(l|l')$ denotes the probability that motion action a, when executed at l', carries the robot to l. $p_a(l|l')$ is used to update the belief upon robot motion:

$$p(L_{t+1} = l) \leftarrow \frac{\sum_{l'} p_a(l|l') p(L_t = l)}{p(s)} \qquad (1)$$

   Here $p(s)$ is a normalizer that ensures that $p(L_{t+1})$ sums up to 1 over all l. An example of $p_a(l|l')$ is shown in Figure 6, considering that the robot moves in the thick roadmap, and the orientation of the robot is aligned to one of the possible 8 directions θ_i, as it was described in a previous section.

2. ... the robot senses. When sensing s,

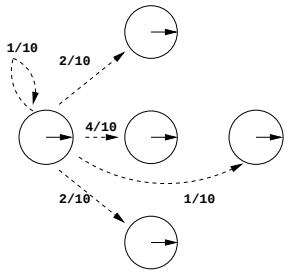


Figure 6: Robot motion for one direction

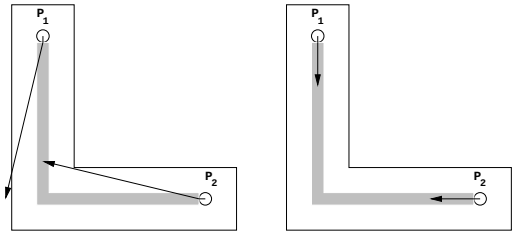


Figure 7: Hypotheses after a movement. From left to right: (a) One hypotheses. (b) Two hypothesis

$$p(L_{t+1} = l) \leftarrow \frac{p(s|l)p(L_t = l)}{p(s)} \qquad (2)$$

Here $p(s|l)$ is the probability of perceiving s at location l. In order to get a efficient procedure to update the probability distribution, cells with probability below some threshold are set to zero.

To get a robust and efficient procedure, our approach considers a *thick* roadmap which include the cells in the neighborhood of the roadmap (*thin roadmap*) besides the cells in the roadmap. The possible hypotheses for the robot are restricted to be in the thick roadmap.

5 Hypothesis elimination

Let $p(L_t = l)$ be the probability of finding the robot at location l at time t. We consider as hypotheses only those locations l with $p(L_t = l) > \beta$, where β is a threshold value. Let n be the number of hypotheses.

To eliminate hypotheses, the robot should move through the roadmap. After a robot move, there are two different scenarios for the cells that correspond to the different hypotheses: 1) the cells are in the roadmap under all the hypotheses, and 2) the cells are in the roadmap for less than n hypotheses. In the first scenario, the movement does not help to eliminate hypotheses; in the second scenario the movement helps to decrease the number of hypotheses. Figure 7 shows an example of both scenarios. If there are two hypotheses P_1 and P_2 then the movement, *represented by a rotation followed by a translation*, showed in Figure 7 (a) helps to reduce the number of hypotheses while the other movement is not useful.

To get an efficient procedure, our approach considers that the mobile robot is at the most probable location and then considers all the cells of the thin roadmap as valid movements for the robot. If we assign to locations the number of possible hypotheses, a good movement to eliminate hypotheses is to direct the robot toward the nearest cell with less than n hypotheses.

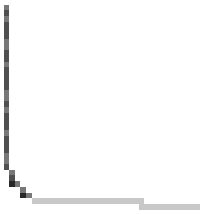


Figure 8: Number of hypotheses associated to cells of the roadmap using the roadmap criteria. Darker cells have lower number of hypotheses

If we consider the hypotheses shown in Figure 5, Figure 8 shows the number of possible hypothesis as an image. Darker cells have lower values for the number of hypotheses. In this case the best hypothesis is located in the right bottom part of the roadmap and it is pointing to the left. The robot should move to the nearest darker cell of the roadmap. Let this cell be called the $goal$ cell.

If we have a criteria to consider two local map views as *similar*, we can use it to detect shorter movements than using only the previous criteria. The two scenarios discussed have two variants: a) the cells are *similar* considering its local map views, and b) not all the cells look similar. The robot should move to a cell with scenario (b).

A similarity measure can be computed using the correlation technique previously presented. Two cells look similar if the computed probability is above some threshold. If there are m cells in the roadmap, let c_i, $(i = 1, .., m)$ denote the m cells of the roadmap, and $sim(c_i, c_j)$ be the similarity between the local map view associated to cells c_i and c_j. $sim(c_i, c_j)$ for all $i, j = 1, ..., m$ form a *similarity matrix* and it can be computed from the map and the roadmap, before the localization process starts. To compute this matrix we can take advantage of the fact that this matrix is symmetric and have zero values in the main diagonal.

Figure 9 shows the similarity matrix for our example. In this case, the cells of the thin roadmap were ordered following a depth first search trying to keep cells with adjacent indices associated to adjacent cells in the roadmap. The image shows that there are only a few cells with significant similarities that are not close to each other. This makes sense if we consider the map shown in Figure 1 and the limited range of the sensors.

Two cells c'_i and c'_j in the thick roadmap are considered similar if $sim(c_i, c_j)$ is above some threshold, where c_i and c_j are the closest cells in the thin roadmap to c'_i and c'_j respectively.

If we consider the hypotheses shown in Figure 5 and the similarity criteria, Figure 10 shows the number of possible hypotheses as an image. In this case, the goal cell is closer to the starting location than the one in Figure 8.

Another improvement over the last criteria is to cluster the

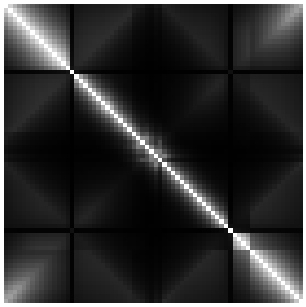


Figure 9: Similarity matrix. Darker pixels mean lower similarities

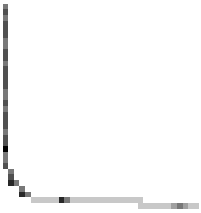


Figure 10: Number of hypotheses associated to cells of the roadmap using the roadmap and the similarity criteria. Darker cells have lower number of hypotheses

hypotheses and then the goal cell is the cell that reduces the number of clusters. In other words, the best movement for the robot should reduce the number of clusters, not the total number of hypotheses. For instance, if there are two clusters (c_1, c_2) and c_1 has 4 hypotheses and c_2 has only 1 hypothesis, and the most probable hypothesis is in cluster c_1, then a movement that eliminates one hypothesis in cluster c_2 is better than one movement that eliminates one or two hypotheses in cluster c_1.

For the clustering process we use Autoclass with the coordinates of the hypotheses. If there is only one cluster, then the goal cell is the nearest cell that reduces the number of hypotheses. The robot is localized if there is only one cluster and there is no cell that reduces the number of hypotheses.

Once a goal cell is found, the robot should perform some movements towards that cell, updating the probability distribution.

6 Experimental Results

This section presents preliminary results obtained using a mobile robot simulator and a real mobile robot. Both robots use (or simulate) sonars and a low cost laser range sensor, implemented with a laser line generator and a camera. The laser sensor gives good measurements within a range of 3 m.

Figure 11 (a) shows a more complex simulated environment. Figure 11 (b) shows the map built with its roadmap. The map is about 10x10 m^2 and the roadmap considers a distance to obstacles of 1m.

Figure 12 shows 7 clusters found by Autoclass, and Figure 13 shows the associated similarity matrix. The symmetric nature of the map is captured in the similarity matrix.

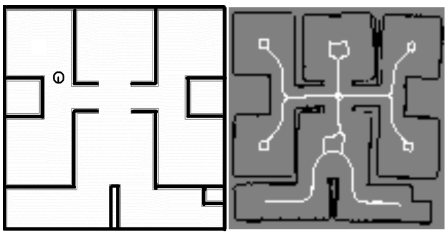


Figure 11: A more complex map built using a mobile robot simulator. From left to right: (a) Simulated environment. (b) Map built with its roadmap

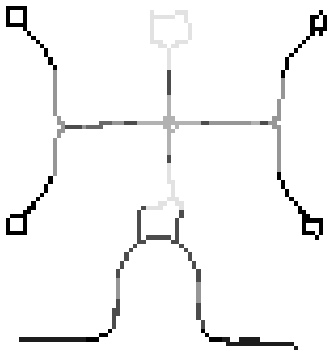


Figure 12: Clusters found by Autoclass for environment of the Figure 11.

If the robot is located as it is shown in Figure 11 (a), then Figure 14 shows the most probable locations for the robot. Autoclass found two clusters C_1 (pointing downwards) and C_2 (pointing upwards) and the location with the highest probability is in C_1. The hypotheses elimination process found a goal cell pointed by G in the figure. This result makes sense because the possible hypotheses after that movement have different sensor views.

Figure 15 shows a map built using the real mobile robot. Autoclass found in this case 6 clusters and the similarity matrix is shown in Figure 15. In this real environment the robot localization is easier than in the previous simulated environ-

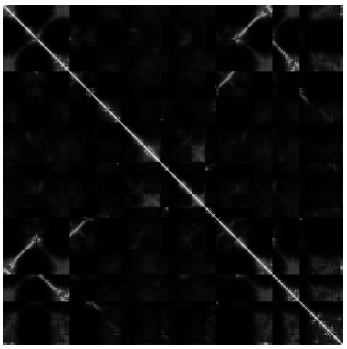


Figure 13: Similarity matrix for environment of the Figure 11. Darker pixels mean lower similarities

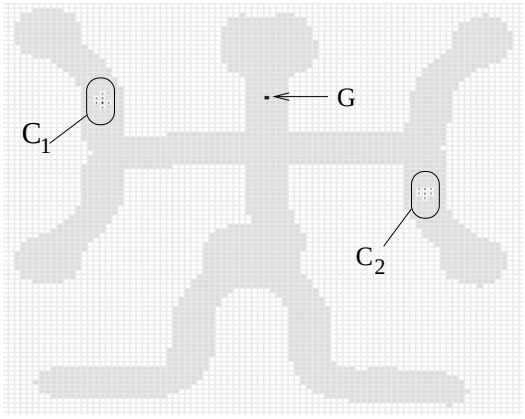


Figure 14: Most probable locations for the robot for the environment showed in Figure 11 (a).

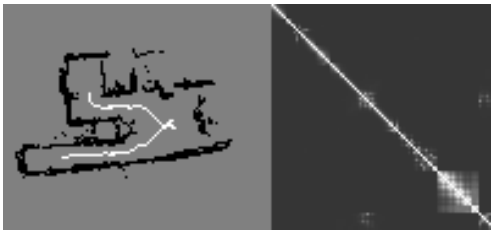


Figure 15: A real mobile robot in a real environment. From left to right: (a) Map and its roadmap. (b) Similarity matrix

ment.

7 Conclusions

An hybrid approach to solve the global localization problem in indoor environments has been presented. It can be seen as a merge of a landmark approach and a dense sensor method based on the Markov localization. Landmarks are generated and recognized by a Bayesian clustering technique (Autoclass) from rotation invariant features (principal components) of sensor or local map views. These landmarks are used to restrict the set of possible locations for the robot and so increase the speed of the Markov localization process.

In the hypothesis elimination phase a fast greedy search is introduced. This search uses a similarity matrix to compute a similarity criteria between the map views of cells. The search considers movements towards cells of the thin roadmap that reduce the number of clusters of hypotheses.

To get an efficient localization algorithm, the approach uses the thick roadmap concept, instead of the full set of free cells; restricts the number of possible orientations of the robot, takes advantage of a correlation technique to align the robot and pre–compute the similarity matrix.

We plan to explore a non deterministic kind of movement in the hypotheses elimination phase using the Markov framework to perform sequence of (move–sense) for each cluster of hypotheses. We think this kind of movements will be able to predict a goal cell even when the map is not very accurate or in the case of robots with high odometric errors.

References

[Borenstein *et al.*, 1996] J. Borenstein, B. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques*. A.K. Peter, Ltd., Wellesley, MA, 1996.

[Castellanos and Tardos, 1999] J.A. Castellanos and J. D. Tardos. *Mobile robot localization and map building: a multisensor fusion approach*. Kluwer Academic Publishers, 1999.

[Cheeseman and Stutz, 1996] P. Cheeseman and J Stutz. Bayesian classification (autoclass): Theory and results. In U.-M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*. AAAI Press/MIT Press, 1996.

[Dudek *et al.*, 1998] G. Dudek, K. Romanik, and S. Whitesides. Localizing a robot with minimum travel. *SIAM J. Comput*, 27(2):583–604, 1998.

[Fox and Burgard, 1998] D. Fox and W. Burgard. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 1998.

[Gonzalez and Wintz, 1987] R. C. Gonzalez and P. Wintz. *Digital Image Processing*. Addison–Wesley, 2 edition, 1987.

[Guibas *et al.*, 1995] L. Guibas, R. Motwani, and P. Raghavan. The robot localization problem. In K. Goldberg, D. Halperin, and J. C. Latombe, editors, *Algorithmic Foundations of Robotics*. A. K. Peters, 1995.

[Gutmann *et al.*, 1998] J.-S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *Proc. International Conference on Intelligent Robots and Systems (IROS'98)*, 1998.

[Romero *et al.*, 2000] L. Romero, E. Morales, and E. Sucar. A robust exploration and navigation approach for indoor mobile robots merging local and global strategies. In M. C. Monard and J.S. Sichman, editors, *Advances in Artificial Intelligence, LNAI 1952*. Springer, 2000.

[Thrun *et al.*, 1998] S. Thrun, A. Bucken, W. Burgar, et al. Map learning and high-speed navigation in rhino. In D. Kortenkamp, R. P. Bonasso, and R Murphy, editors, *Artificial Intelligence and Mobile Robots*. AAAI Press/The MIT Press, 1998.

[Thrun, 1998] S. Thrun. Learning maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

[Weib *et al.*, 1994] G. Weib, C. Wetzler, and E. Puttkamer. Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. In *Intelligent Robots and Systems*, 1994.

Multimodal Integration – A Biological View

Michael H. Coen
MIT Artificial Intelligence Lab
545 Technology Square
Cambridge, MA 02139
mhcoen@ai.mit.edu

Abstract

We present a novel methodology for building highly integrated multimodal systems. Our approach is motivated by neurological and behavioral theories of sensory perception in humans and animals. We argue that perceptual integration in multimodal systems needs to happen at all levels of the individual perceptual processes. Rather than treating each modality as a separately processed, increasingly abstracted pipeline – in which integration over abstract sensory representations occurs as the final step – we claim that integration and the sharing of perceptual information must also occur at the earliest stages of sensory processing. This paper presents our methodology for constructing multimodal systems and examines its theoretic motivation. We have followed this approach in creating the most recent version of a highly interactive environment called the Intelligent Room and we argue that doing so has provided the Intelligent Room with unique perceptual capabilities and gives insight into building similar complex multimodal systems.

Introduction

This paper proposes a novel perceptual architecture motivated by surprising results about how the brain processes sensory information. These results, gathered by the cognitive science community over the past 50 years, have challenged century long held notions about how the brain works and how we experience the world we live in. We argue that current approaches to building multimodal systems that perceive and interact with the real, human world are flawed and based largely upon assumptions described by Piaget (1954) – although tracing back several hundred years – that are no longer believed to be particularly accurate or relevant.

Instead, we present a biologically motivated methodology for designing interactive systems that reflects more of how the brain actually appears to process and merge sensory inputs. This draws upon neuroanatomical, psychophysical, evolutionary, and phenomenological evidence, both to critique modern approaches and to suggest an alternative for building artificial perceptual systems. In particular, we argue against *post-perceptual* integration, which occurs in systems where the modalities are treated as separately processed, increasingly abstracted pipelines. The outputs of these pipelines are then merged in a final integrative step, as in Figure 1. The main difficulty with this approach is that integration happens after the individual perceptions are generated. Integration occurs *after* each perceptual subsystem has already "decided" what it has perceived, when it is too late for intersensory influence to affect the individual, concurrent unimodal perceptions. Multimodal integration is thus an *assembly* rather than a *perceptual* process in most modern interactive systems. In this, it is an abstraction mechanism whereby perceptual events are

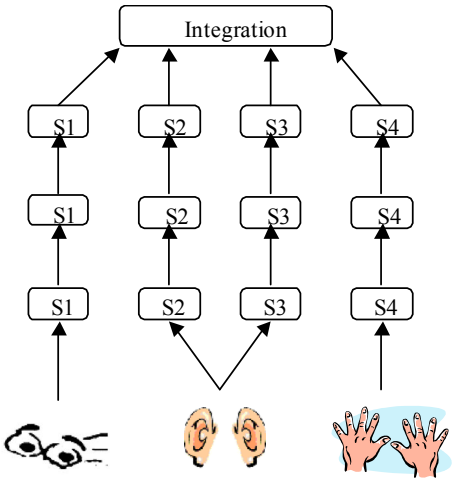


Figure 1 – Post-perceptual integration in a multimodal system. Here, visual, auditory, and haptic inputs pass through specialized unimodal processing pathways and are combined in the integration mechanism on top, which creates multimodal perceptions by extracting and reconciling data from the individual inputs. Notice that multiple pathways may process a single sensory input. For example, the auditory signal in this system might be fed into separate *speech recognition* and *spatial localization systems*, designated S2 and S3 respectively.

This material is based upon work supported by the Advanced Research Projects Agency of the Department of Defense under contract number DAAD17-99-C-0060, monitored through Rome Laboratory. Additional support was provided by the Microsoft Corporation.

separated from the specific sensory mechanisms that generate them and then integrated into higher-level representations.

This paper examines the idea that multimodal integration is a fundamental component of perception itself and can shape individual unimodal perceptions as much as does actual sensory input. This idea is well supported biologically, and we argue here for the benefits of building interactive systems that support *cross-modal influence* – systems in which sensory information is shared across all levels of perceptual processing and not just in a final integrative stage. Doing this, however, requires a specialized approach that differs in basic ways from how interactive systems are generally designed today.

This paper presents our methodology for constructing multimodal systems and examines its theoretic motivation. We have followed this approach in creating the most recent version of a highly interactive environment called the Intelligent Room, and we argue that doing so has provided the Intelligent Room with unique perceptual capabilities and provides insight into building similar complex multimodal systems. We specifically examine here how the Intelligent Room's vision and language systems share information to augment each other and why the traditional ways these systems are created made this sharing so difficult to implement. Our approach is similar in spirit to the work of (Atkeson et al. 2000, Brooks et al. 1998, Cheng and Kuniyoshi 2000, Ferrell 1996, Nakagawa 1999, and Sandini et al. 1997). Although they are primarily concerned with sensorimotor coordination, there is a common biological inspiration and long-term goal to use knowledge of human and animal neurophysiology to design more sophisticated artificial systems.

Background

Who would question that our senses are distinct? We see, we feel, we hear, we smell, and we taste, and these are qualitatively such different experiences that there is no room for confusion among them. Even those affected with the peculiar syndrome *synesthesia*, in which real perceptions in one sense are accompanied by illusory ones in another, never lose awareness of the distinctiveness of the senses involved. Consider the woman described in (Cytowic 1988), for whom a particular taste always induced the sensation of a particular geometric object in her left hand. A strange occurrence indeed, but nonetheless, the tasting and touching – however illusory – were never confused; they were never merged into a sensation the rest of us could not comprehend, as would be the case, for example, had the subject said something *tasted* octagonal. Even among those affected by synesthesia, the sensory channels remain extremely well defined.

Given that our senses appear so unitary, how does the brain coordinate and combine information from different sensory modalities? This has become known as the *binding problem,* and the traditional assumption has been to assume that the sensory streams are abstracted, merged, and integrated in the cortex, at the highest levels of brain functioning. This was the solution proposed by Piaget (1954) and in typical Piagetian fashion, assumed a cognitive developmental process in which children slowly developed high-level mappings between innately distinct modalities through their interactions with the world. This position directly traces back to Helmholtz (1856), and even earlier, to Berkeley (1709) and Locke (1690), who believed that neonatal senses are congenitally separate and interrelated only through experience. The interrelation *does not diminish the distinctiveness of the senses themselves*, it merely accounts for correspondences among them based on perceived cooccurrences.

The Piagetian assumption underlies nearly all modern interactive, multimodal systems – it is the primary architectural metaphor for multimodal integration. The unfortunate consequence of this has been making integration a post-perceptual process, which assembles and integrates sensory input after the fact, in a separate mechanism from perception itself. In these multimodal systems, each perceptual component provides a generally high-level description of what it sees, hears, senses, etc. (Some systems, such as sensor networks, provide low-level feature vectors, but the distinction is not relevant here.) This, for example, may consist of locations of people and how they are gesturing, what they are saying and how (e.g., prosody data), and biometric data such as heart rates. This information is conveyed in modal-specific representations that capture detail sufficient for higher-level manipulation of the perceptions, while omitting the actual signal data and any intermediate analytic representations. Typically, the perceptual subsystems are independently developed and trained on unimodal data; each system is designed to work in isolation. (See Figure 2.) They are then interconnected through some fusive mechanism (as in Figure 1) that combines temporally proximal, abstract unimodal inputs into an integrated event model. The integration itself may be effected via a neural network (e.g., Waibel et al. 1995), hidden Markov models (e.g., Stork and Hennecke 1996), unification logics (e.g., Cohen et al. 1997), or various ad hoc techniques (e.g., Wu et al. 1999). The output of the integration process is then fed into some higher-level interpretative mechanism – the architectural equivalent of a cortex.

This post-perceptual approach to integration denies the possibility of cross-modal influence, which is pervasive in biological perception. Our visual, auditory, proprioceptive, somatosensory, and vestibular systems influence one another in a complex process from which perceptions emerge as an integrated product of a surprising diversity of components. (For surveys, see Stein and Meredith 1993, Lewkowicz and Lickliter 1994, and to a lesser extent, Thelen and Smith 1994.) For example, consider the seminal work of McGurk and MacDonald (1976). In preparing an experiment to determine how infants reconcile conflicting information in different sensory modalities, they had a lab technician dub

the audio syllable /ba/ onto a video of someone saying the syllable /ga/. Much to their surprise, upon viewing the dubbed video, they repeatedly and distinctly heard the syllable /da/ (alternatively, some hear /tha/), corresponding neither to the actual audio nor video sensory input. Initial assumptions that this was due to an error on the part of the technician were easily discounted simply by shutting their eyes while watching the video; immediately, the sound changed to a /ba/. This surprising fused perception, subsequently verified in numerous redesigned experiments and now known as the *McGurk effect*, is robust and persists even when subjects are aware of it.

The McGurk effect is perhaps the most convincing demonstration of the intersensory nature of face-to-face spoken language and the undeniable ability of one modality to radically change perceptions in another. It has been one of many components leading to the reexamination of the Piagetian introspective approach to perception. Although it may seem reasonable to relegate intersensory processing to the cortex for the *reasoning* (as opposed to *perceptual*) processes that interested Piaget, such as in cross modal matching, it becomes far more implausible in cases where different senses impinge upon each other in ways that locally change the perceptions in the sensory apparatus themselves. One might object that the McGurk effect is pathological – it describes a perceptual phenomenon outside of ordinary experience. Only within controlled, laboratory conditions do we expect to have such grossly conflicting sensory inputs; obviously, were these signals to co-occur naturally in the real world, we would not call them conflicting. We can refute this objection both because the real world *is* filled with ambiguous, sometimes directly conflicting, perceptual events, and because the McGurk effect is by no means the only example of its kind. There is a large and growing body of evidence that the type of direct perceptual influence illustrated by the McGurk effect is commonplace in much of ordinary human and more generally animal perception, and it strongly makes the case that our perceptual streams are far more interwoven than conscious experience tends to make us aware. For example, the sight of someone's moving lips in an environment with significant background noise makes it easier to understand what the speaker is saying; *visual* cues – e.g., the sight of lips – can alter the signal-to-noise ratio of an *auditory* stimulus by 15-20 decibels (Sumby and Pollack 1954). Thus, a decrease in auditory acuity can be offset by increased reliance on visual input. Although the neural substrate behind this interaction is unknown, it has been determined that just the *sight* of moving lips – without any audio component – modifies activity in the *auditory* cortex (Sams et al 1991). In fact, psycholinguistic evidence has long lead to the belief that lip-read and heard speech share a degree of common processing, notwithstanding the obvious differences in their sensory channels (Dodd et al 1984).

Perhaps the most dramatic cross-modal interactions were demonstrated in the landmark studies of Meltzoff and Moore (1977), who showed that infants could imitate an investigator's facial expression within hours of birth. For example, the investigator sticking out his tongue lead the infant to do the same, although the infant had never seen its own face. Somehow, the visual cue is matched with the proprioceptive sensation of tongue protrusion. It is extraordinarily difficult to view this as a learned behavior; this will be quite relevant in considering below where the knowledge governing cross-modal influence should come from in computational systems.

We believe that the current approach to building multimodal interfaces is an artifact of how people like to build computational systems and not at all well-suited to dealing with the cross-modal interdependencies of perceptual understanding. Perception does not seem to be amenable to the clear-cut abstraction barriers that computer scientists find so valuable for solving other problems, and we claim this approach has lead to the fragility of so many multimodal systems. We also dispute the notion that perception generally corresponds well to a discrete and symbolic *event* model. Although such a model is well suited to many types of computational applications, it is frequently more useful to view perception as a dynamic process, rather than an event occurring at some point in time (Thelen and Smith 1994). This is all the more so during development (i.e. learning), where the space of events is fluid and subject to constant change, and during sensory fusion, where complex codependences and interactions among the different senses confound our simple event-biased predispositions. A similar dynamic approach has been taken by (Ullman 1996) for explaining cross-feature influence in visual perception.

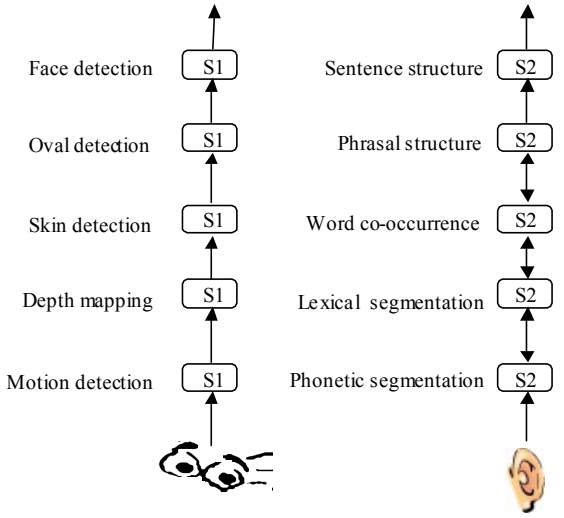


Figure 2 – Unimodal processing pathways. Individual modalities are processed in specialized pipelines. The visual pathway on the left uses 3-dimensional depth and color maps to find candidate regions for locating faces. (This follows Darrell et al. 1998.) The auditory pathway on the right performs speech recognition. Notice that higher-level syntactic constraints feed back into the lower-level morphological analysis.

A Multimodal System

The examples for our discussion of multimodal interactions will be drawn from the Intelligent Room project, as described in (Coen 1998, 1999). The Intelligent Room has multiple perceptual user interfaces, supporting both visual and verbal interactions, which connect with some of the ordinary human-level events going on within it. The goal of the room is to support people engaged in everyday, traditionally non-computational activity in both work and leisure contexts. Figure 3 contains a picture of the room, which contains nine video cameras, three of which the room can actively steer, several microphones, and a large number of computer-interfaced devices. Because the Intelligent Room is designed to support a range of activities that have never before explicitly involved computers, it in no way resembles a typical computer science laboratory. Its computational infrastructure is removed from view to enhance the room's appeal as a *naturally* interactive space with a decidedly understated "low-tech" atmosphere. The room's computational infrastructure (Coen et al. 1999), consisting of over 120 software agents running on a network of ten workstations, is housed in an adjacent laboratory.

Before exploring scenarios for new types of multimodal interactions, we first examine a traditionally explicit one in the resolution of a deictic reference, i.e., use of a word such as *this* in referring to a member of some class of objects. Suppose, for example, someone in the room says, "Dim this lamp." The room uses its ability to track its occupants, in conjunction with a map of its own layout, to dim the lamp *closest* to the speaker when the verbal command was issued. This kind of interaction can be implemented with a simple post-perceptual integration mechanism that reconciles location information obtained from the person tracker with the output of a speech recognition system. Here, multimodal integration of positional and speech information allows straightforward disambiguation of the deictic lamp reference.

Motivations

Given the simplicity of the above example, it seems far from obvious that a more complex integration mechanism is called for. To motivate a more involved treatment, we start by examining some of the problems with current approaches.

Despite many recent and significant advances, computer vision and speech understanding, along with many other perceptual interface research areas (Picard 1997, Massie and Salisbury 1994), are still infant sciences. The non-trivial perceptual components of multimodal systems are therefore never "perfect" and are subject to a wide variety of failure modes. For example, the room may "lose" people while visually tracking them due to occlusion, coincidental color matches between fore and background objects, unfavorable lighting conditions, etc. Although the particular failure

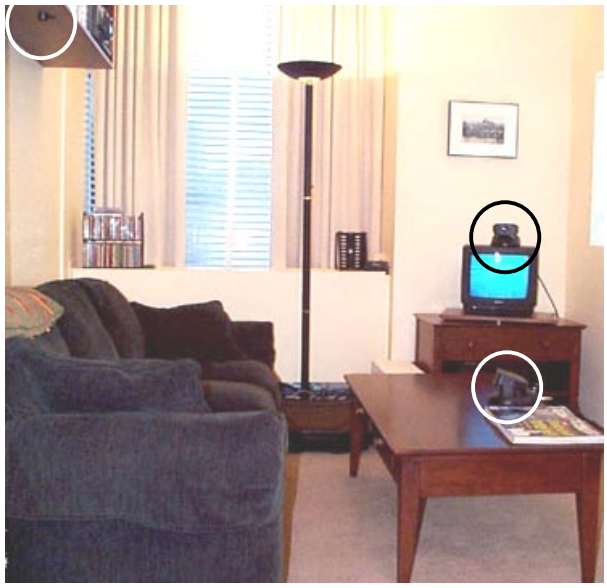


Figure 3 – A view of the Intelligent Room with three of its nine cameras visible and circled. The two lower of these in the picture can be panned and tilted under room control.

modes of the modalities varies with them individually, it is a safe assumption that under a wide variety of conditions any one of them may temporarily stop working as desired. How these systems manifest this undesired operation is itself highly idiosyncratic. Some may simply provide no information, for example, a speech recognition system confused by a foreign accent. Far more troublesome are those that continue to operate as if nothing were amiss but simply provide incorrect data, such as a vision-based tracking system that mistakes a floor lamp for a person and reports that he is standing remarkably still.

That perceptual systems have a variety of failure modes is not confined to their artificial instantiations. Biological systems also display a wide range of pathological conditions, many of which are so engrained that they are difficult to notice. These include limitations in innate sensory capability, as with visual blind spots on the human retina, and limited resources while processing sensory input, as with our linguistic difficultly understanding nested embedded clauses (Miller and Chomsky 1963). Stein and Meredith (1994) argue for the evolutionary advantages of overlapping and reinforcing sensory abilities; they reduce dependence on specific environmental conditions and thereby provide clear survival advantages. A striking example of this is seen in a phenomenon known as the "facial vision" of the blind. In locating objects, blind people often have the impression of a slight touch on their forehead, cheeks, and sometimes chest, as though being touched by a fine veil or cobweb (James 1890, p204). The explanation for this extraordinary perceptory capability had long been a subject of fanciful debate. James demonstrated, by stopping up the ears of blind subjects with putty, that audition was behind this sense, which is now known to be caused by

intensity, direction, and frequency shifts of reflected sounds (Arias 1996). The auditory input is so successfully represented haptically in the case of facial vision that the perceiver himself cannot identify the source of his perceptions.

Research on interactive systems has focused almost entirely on unimodal perception: the isolated analysis of auditory, linguistic, visual, haptic, or to a lesser degree biometric data. It seems to put the proverbial cart before the horse to ponder how information from different modalities can be merged while the perceptory mechanisms in the sensory channels are themselves largely unknown. Is it not paradoxical to suggest we should or even could study integration without thoroughly understanding the individual systems to be integrated? Nonetheless, that is the course taken here. We argue that while trying to understand the processing performed within individual sensory channels, *we must simultaneously ask how their intermediary results and final products are merged into an integrated perceptual system*. We believe that because perceptual systems within a given species coevolved to interoperate, *compatibility pressures existed on their choices of internal representations and processing mechanisms*. In order to explain the types of intersensory influence that have been discovered experimentally, *disparate perceptual mechanisms must have some degree of overall representational and algorithmic compatibility that makes this influence possible*. The approach taken here is entirely gestalt, not only from a Gibsonian (1986) perspective, but because we have no example of a complex unimodal sensory system evolving in isolation. Even the relatively simple perceptual mechanisms in paramecium (Stein and Meredith 1994, Chapter 2) and sponges (Mackie and Singla 1983) have substantial cross-sensory influences. It seems that perceptual interoperation is a prerequisite for the development of complex perceptual systems. Thus, rather than study any single perceptual system in depth – the traditional approach – we prefer to study them *in breadth*, by elucidating and analyzing interactions between different sensory systems.

Cross-Modal Influences

How then might cross-modal influences be used in a system like the Intelligent Room? Answering this question is a two-step process. Because the Intelligent Room is an engineered as opposed to evolved system, we first need to explicitly find potential synergies between its modalities that can be exploited. Once determined, these synergies must then somehow be engineered into the overall system, and this emerges as the primary obstacle to incorporating cross-modal influences into the Intelligent Room and more generally, to other types of interactive systems.

We begin with the following two empirical and complementary observations:

1) People tend to talk about objects they are near. (Figure 4a)

2) People tend to be near objects they talk about. (Figure 4b)

These heuristics reflect a relationship between a person's location and what that person is referring to when he speaks; knowing something about one of them provides some degree of information about the other. For example, someone walking up to a video display of a map is potentially likely to speak about the map; here, person location data can inform a speech model. Conversely, someone who speaks about a displayed map is likely to be in a position to see it; here, speech data can inform a location model. Of course, it is easy to imagine situations where these heuristics would be wrong. Nonetheless, as observations they are frequently valid and it would be reasonable to somehow incorporate influences based on them into a system like the Intelligent Room. Mechanistically, we might imagine the person tracking system exchanging information with the speech recognition system. For example, the tracking system might send data, which we will call a *hint,* to the speech recognition system to preferentially expect utterances involving objects the person is near, such as a map. Conversely, we can also imagine that the speech recognition

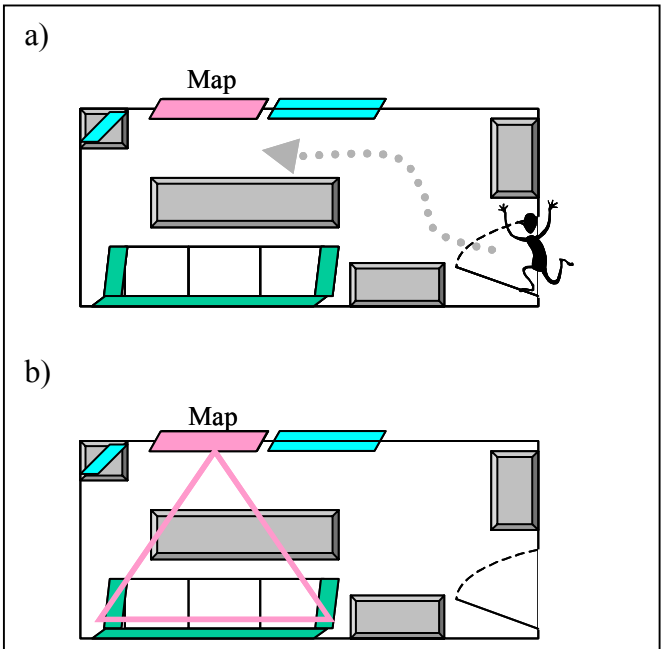


Figure 4a – People talk about objects they are near. Someone approaching a projected display showing, for example, a map, is more likely to make a geographical utterance. Here, location information can augment speech recognition.

Figure 4b – People are near objects they talk about. Someone speaking about the contents of a video display is more likely to be located somewhere (delineated by the triangle) from which the display is viewable to him. Here, speech information can augment person tracking.

system would send hints to the person tracking system to be especially observant when looking for someone in indicated sections of the room, based on what that person is referring to in his speech.

This seems reasonable until we try to build a system that actually incorporates these influences. There are both representational and algorithmic stumbling blocks that make this conceptually straightforward cross-modal information sharing difficult to implement. These are due not only to post-perceptual architectural integration, but also to how the perceptual subsystems (such as those in Figure 2) are themselves typically created. We first examine issues of representational compatibility, namely what interlingua is used to represent shared information, and then address how the systems could incorporate *hints* they receive in this interlingua into their algorithmic models.

Consider a person tracking system that provides the coordinates of people within a room in real-time, relative to some real-world origin – the system outputs the actual locations of the room's occupants. We will refer to the tracking system in the Intelligent Room as a representative example of other such systems (e.g., Wren et al. 1997, Gross et al. 2000). Its only input is a stereo video camera and its sole output are sets of *(x,y,z)* tuples representing occupants' centroid head coordinates, which are generated at 20Hz. Contrast this with the Intelligent Room's speech recognition system, which is based upon the Java Speech API (Sun 2001), built upon IBM's ViaVoice platform, and is typical of similar spoken language dialog systems (e.g., Zue et al. 2000). Its inputs are audio voice signals and a formal linguistic model of expected utterances, which are represented as probabilistically weighted context free grammars.

How then should these two systems exchange information? It does not seem plausible from an engineering perspective, whether in natural or artificial systems, to provide each modality with access to the internal representations of the others. Thus, we do not expect that the tracking system should know anything about linguistic models nor we do expect the language system should be skilled in spatial reasoning and representation. Even if we were to suppose the speech recognition system *could* somehow represent spatial coordinates, e.g. as *(x,y,z)* tuples, that it could communicate to the person tracking system, the example in Figure 4b above involves *regions* of space, not isolated point coordinates. From an external point of view, it is not obvious how the tracking system internally represents regions, presuming it even has that capability in the first place. The complementary example of how the tracking system might refer to classes of linguistic utterances, as in Figure 4a above, is similarly convoluted.

Unfortunately, even if this interlingua problem were easily solvable and the subsystems had a common language for representing information, the way most perceptual subsystems are implemented would make incorporation of cross-modal data difficult or impossible. For example, in the case of a person tracking system, the real-world body coordinates are generated via three-dimensional spatial reconstruction based on correspondences between sets of image coordinates. The various techniques for computing the reconstructed coordinates, such as neural networks or fit polynomials, are in a sense closed – once the appropriate coordinate transform has been learned, there is generally no way to bias the transformation in favor of particular points or spatial regions. Thus, there is no way to incorporate the influence, even if the systems had a common way of encoding it. Here again, the complementary situation with influencing speech recognition from a tracking system can be similarly intractable. For example, not all linguistic recognition models (e.g., bigram-based) support dynamic preferential weighting for classes of commonly themed utterances. So, even if the tracking system could somehow communicate what the speech recognition system should expect to hear, the speech recognition system might not be able to do anything useful with this information.

We see that not only are the individual modal representations incompatible, the perceptual algorithms (i.e., the contents of the sensory pipelines) are incompatible as well. This comes as no surprise given that these systems were engineered primarily for unimodal applications. Unlike natural perceptual systems within an individual species, artificial perceptual systems do not co-evolve, and therefore, have had no evolutionary pressure to force representational and algorithmic compatibility. These engineered systems are intended to be data sources feeding into other systems, such as the ones performing multimodal integration, that are intended to be data sinks. There is no reason to expect that these perceptual subsystems would or even could directly interoperate.

Designing for Interaction

Our solution was to redesign the Intelligent Room's perceptual systems with the explicit intension that they should interact with each other. Doing so required fundamental representational and algorithmic changes but has made possible subtle types of cross-modal interactions that were previously unworkable. We first detail two different categories of intersensory function and then explain how the cross-modal influences described above were implemented in the Intelligent Room.

Consider, for example, the effect of touching someone and having his head and eyes turn to determine the source of the stimulus. This is clearly an example of cross-modal influence – the position of the touch determines the foveation of the eyes – but it is fundamentally different than the interaction described in the McGurk effect above, where the influence is evidenced solely in perceptual channels. The touch scenario leads to behavioral effects that center the stimulus with respect to the body and peripheral sensory organs. The McGurk effect is an example of sensory influence within perceptual channels and has no behavioral component.

Motor influences – i.e., ones that cause attentive and orientation behaviors – are by far the more understood of the

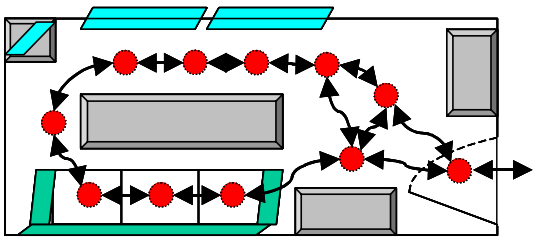


Figure 5 – A learned topographic map overlaid on top of the room's floor plan. The dots represent locations in the room that were used as reference points to interactively train a multi-camera vision system. The arrows represent observed transitions between these reference locations. Once trained, the room can then locate people at and between these points.

two. The primary neurological substrate behind them is the *superior colliculus,* a small region of the brain that produces signals that orient peripheral sensory organs based on sensory stimuli. The superior colliculus contains layered, topographic sensory and motor maps that are in register; that is, co-located positions in the real world – in the sensory case representing derived locations of perceptual inputs and in the motor case representing peripheral sensory organ motor coordinates that focus on those regions – are all essentially vertically overlapping. The actual mechanisms that use these maps to effect intersensory influence are currently unknown – variants on spreading vertical activation are suspected – but there is little doubt the maps' organization is a fundamental component of that mechanism.

Far less is known neurologically about purely semantic influences – i.e., ones that have effects confined to perceptual channels. The superior colliculus itself has been directly approachable from a research perspective because the brain has dedicated inner space, namely, the tissue of the topographic maps, to representing the outer space of the real-world; the representation is both isomorphic and perspicacious, and it has made the superior colliculus uniquely amenable to study. The perceptual as opposed to spatial representations of the senses are far more elusive and are specialized to the individual modalities and the organs that perceive them.

We have used the notion of layered topographic maps to represent both motor *and* semantic information in the Intelligent Room. Even though the superior colliculus has not yet been identified as a substrate in *non-behavioral* cross-modal influences, its extensive intra-map connections to higher cortical areas – particularly the visual cortex – may indicate its role in other types of intersensory function that are confined to perceptual channels and have no behavioral component.

Using a topographic organization, we created a new model for visually tracking people within a room (Coen and Wilson 1999). The model takes advantage of the observation that much of the information needed in human-computer interaction is qualitative in nature. For example, it may be necessary to distinguish between a person sitting on a chair, a person standing in front of a bookcase, and a person standing in a doorway, but obtaining the actual real-world coordinates of these people is generally unimportant. In our system, locations in a room that are likely to contain people are used as reference points, as in Figure 5, to interactively train a multi-camera vision system, whose current implementation has three steerable and six fixed cameras. The system learns to combine event predictions from the multiple video streams in order to locate people at these reference points in the future. The system can also dynamically track people with the steerable cameras as they move between these locations. Because most rooms have natural attractors for human activity, such as doorways, furniture, and displays, the selection of training points is usually readily apparent from the layout of the room.

Once this topographic map is created, the tracking system activates locations on the map to correspond to its observations of people in the room. As we will see in a moment, other systems in the room can also *weakly* activate locations on the map, *which causes the room to turn a steerable camera to view the corresponding real world location.* If a person is found there as a result of orienting the camera, the system then completely activates that location on the map.

The ability to interact with topographic representations of the room is not confined to the tracking system. Once the map is learned, the room builds corresponding maps to spatially categorize events in its other sensory systems, even if they have no explicit spatial component. For example, speech recognition events are topographically organized on a map dedicated just for that purpose. As utterances are heard for which the room has spatial information (either learned or explicitly provided by its programmers), it activates locations in the speech system's topographic map, which in turn activates locations in other modalities' topographic maps via vertical spreading activation, as shown in Figure 6. Conversely, other systems can *weakly* activate locations on the speech system's topographic map, which causes the speech system to increase the expectation probabilities of utterances associated with that location.

Thus, activations in the map are *bi-directional*: perceptual events in a given modality can directly activate its map locations. Maps locations can also be activated via spreading activation from corresponding positions in other system's topographic maps, *which causes a corresponding change in that modality's perceptual state* — here, these spreading activations cause the secondary system to either look or listen for something. It is this bi-directional activation – through which the systems can react to intersensory stimuli – that has made possible the cross-modal influences that were presented in Figure 4.

Conclusion

This paper has described our approach to incorporating cross-modal influences into the perceptual processing of the

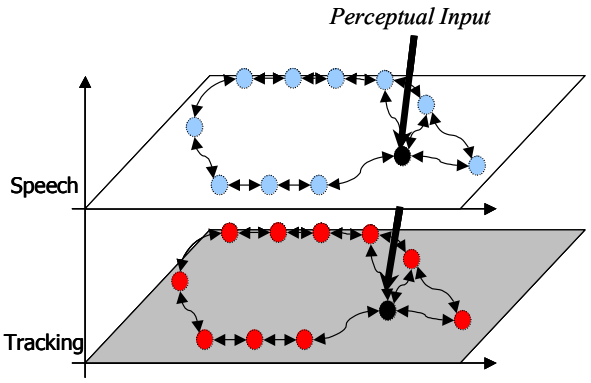


Figure 6 – Layered topographic maps containing location-dependent speech and tracking information. Positions on the maps can be activated directly by perceptual inputs or activated by spreading vertical activation from adjacent maps.

Intelligent Room. We simultaneously argued against conventional post-perceptual integration and have motivated this position with biological evidence that unimodal perceptions are themselves integrated products of multimodal sources. Our position has allowed us to explore representational and algorithmic issues in *unimodal* perception that can only be approached from an integrated, *multimodal* perspective. It has also allowed us to investigate creating more sophisticated interactive systems by incorporating more subtle intersensory cues into the Intelligent Room. Future work is both exciting and promising.

References

1. Atkeson CG, Hale J, Pollick F, Riley M, Kotosaka S, Schaal S, Shibata T, Tevatia G, Vijayakumar S, Ude A, Kawato M: Using humanoid robots to study human behavior. *IEEE Intelligent Systems*, Special Issue on Humanoid Robotics, 46-56. 2000.
2. Brooks, R.A., C. Breazeal (Ferrell), R. Irie, C. Kemp, M. Marjanovic, B. Scassellati and M. Williamson, Alternate Essences of Intelligence. *In Proceedings of The Fifteenth National Conference on Artificial Intelligence.* (AAAI98). Madison, Wisconsin. 1998.
3. Butterworth, G. The origins of auditory-visual perception and visual proprioception in human development. In *Intersensory Perception and Sensory Integration*, R.D. Walk and L.H. Pick, Jr. (Eds.) New York. Plenum. 1981.
4. Cheng, G., and Kuniyoshi, Y. Complex Continuous Meaningful Humanoid Interaction: A Multi Sensory-Cue Based Approach *Proc. of IEEE International Conference on Robotics and Automation* (ICRA 2000), pp.2235-2242, San Francisco, USA, April 24-28, 2000.
5. Coen, M. Design Principles for Intelligent Environments. *In Proceedings of The Fifteenth National Conference on Artificial Intelligence.* (AAAI98). Madison, Wisconsin. 1998.
6. Coen, M. The Future Of Human-Computer Interaction or How I learned to stop worrying and love My Intelligent Room. IEEE Intelligent Systems. March/April. 1999.
7. Coen, M., and Wilson, K. Learning Spatial Event Models from Multiple-Camera Perspectives in an Intelligent Room. In *Proceedings of MANSE'99*. Dublin, Ireland. 1999.
8. Coen, M., Phillips, B., Warshawsky, N., Weisman, L., Peters, S., Gajos, K., and Finin, P. Meeting the computational needs of intelligent environments: The Metaglue System. In *Proceedings of MANSE'99*. Dublin, Ireland. 1999.
9. Cohen, P. R., Johnston, M., McGee, D., Smith, I. Oviatt, S., Pittman, J., Chen, L., and Clow, J. QuickSet: Multimodal interaction for simulation set-up and control. 1997, *Proceedings of the Applied Natural Language Conference,* Association for Computational Linguistics. 1997.
10. Cytowic, R. E., Synesthesia: A Union of Senses. New York. Springer-Verlag. 1989.
11. Darrell, T., Gordon, G., Harville, M., and Woodfill, J., Integrated person tracking using stereo, color, and pattern detection, *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR '98),* pp. 601-609, Santa Barbara, June 1998.
12. Ferrell, C. Orientation behavior using registered topographic maps. In Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (SAB-96). Society of Adaptive Behavior. 1996.
13. Gross, R., Yang, J., and Waibel, A. Face Recognition in a meeting room. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, Grenoble, France, March 2000
14. Held, R. Shifts in binaural localization after prolonged exposures to atypical combinations of stimuli. Am. J. Psychol. 68L526-266. 1955.
15. Helmholtz, H. v. *Handbook of Physiological Optics.* 1856. as reprinted. in James P.C. Southall. (Ed.) 2000.
16. James, H. 1890. Principles of Psychology. Vol. 2. Dover. 1955.
17. Kohler, I. The formation and transformation of the perceptual world. Psychological Issues 3(4):1-173. 1964.
18. McGurk, H., and MacDonald, J. Hearing lips and seeing voices. Nature. 264:746-748. 1976.
19. Meltzoff, A.N. and Moore, M.K. Imitation of facial and manual gestures by human neonates. Science 198:75-78. 1977.
20. Miller, George and Noam Chomsky (1963). Finitary models of language users. In Luce, R.; Bush, R. and Galanter, E. (eds.) *Handbook of Mathematical Psychology, Vol 2*. New York: Wiley. 419-93.
21. Piaget, J. *Construction of reality in the child,* London: Routledge & Kegan Paul, 1954.
22. Sams, M., Aulanko, R., Hamalainen, M., Hari, R., Lounasmaa, O., Lu, S., and Simola, J. Seeing speech: Visual information from lip movements modified activity in the human auditory cortex. Neurosci. Lett. 127:141-145. 1991.
23. Sandini G., Metta G. and Konczak J. "Human Sensori-motor Development and Artificial Systems". In: AIR&IHAS '97, Japan. 1997.
24. Stein, B., and Meredith, M. A. The Merging of the Senses. Cambridge, MA. MIT Press. 1994.
25. Stork, D.G., and Hennecke, M. Speechreading: An overview of image processing, feature extraction, sensory integration and pattern recognition techniques", *Proc. of the Second Int. Conf. on Auto. Face and Gesture Recog.* Killington, VT pp. xvi--xxvi 1996.
26. Sumby, W.H., and Pollack, I. Visual contribution to speech intelligibility in noise. J. Acoust. Soc. Am. 26:212-215. 1954.
27. Sun. http://www.javasoft.com/products/java-media/speech/. 2001.
28. Thelen, E., and Smith, L. A Dynamic Systems Approach to the Development of Cognition and Action. Cambridge, MIT Press. 1998.
29. Ullman, Shimon. High-level vision: object recognition and visual cognition. Cambridge. MIT Press. 1996.
30. Waibel, A., Vo, M.T., Duchnowski, P., and Manke, S. Multimodal Interfaces. Artificial Intelligence Review. 10:3-4. p299-319. 1996.
31. Wren, C., Azarbayejani, A., Darrell, T., and Pentland, P., "Pfinder: Real-Time Tracking of the Human Body ", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1997.
32. Wu, Lizhong, Oviatt, Sharon L., Cohen, Philip R., Multimodal Integration -- A Statistical View, *IEEE Transactions on Multimedia*, Vol. 1, No. 4, December 1999, pp. 334-341.
33. Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T., and Hetherington, L. "JUPITER: A Telephone-Based Conversational Interface for Weather Information," IEEE Transactions on Speech and Audio Processing, Vol. 8, No. 1, January 2000.

Real-Time Auditory and Visual Multiple-Object Tracking for Humanoids

Kazuhiro Nakadai[†], Ken-ichi Hidai[†], Hiroshi Mizoguchi[‡], Hiroshi G. Okuno[†§], and Hiroaki Kitano [†¶]

† Kitano Symbiotic Systems Project, ERATO, Japan Science and Technology Corp.

Mansion 31 Suite 6A, 6-31-15 Jingumae, Shibuya-ku, Tokyo 150-0001, Japan

‡ Department of Information and Computer Science, Saitama University, Saitama 338-8570, Japan

§ Department of Intelligence Science and Technology, Kyoto University, Kyoto 606-8501, Japan

¶ Sony Computer Science Laboratories, Inc., Tokyo 141-0022, Japan

{nakadai, hidai, okuno, kitano}@symbio.jst.go.jp, hm@me.ics.saitama-u.ac.jp

Abstract

This paper presents a real-time auditory and visual tracking of multiple objects for humanoid under real-world environments. Real-time processing is crucial for sensorimotor tasks in tracking, and multiple-object tracking is crucial for real-world applications. Multiple sound source tracking needs perception of a mixture of sounds and cancellation of motor noises caused by body movements. However its real-time processing has not been reported yet. Real-time tracking is attained by fusing information obtained by sound source localization, multiple face recognition, speaker tracking, focus of attention control, and motor control. Auditory streams with sound source direction are extracted by active audition system with motor noise cancellation capability from 48 KHz sampling sounds. Visual streams with face ID and 3D-position are extracted by combining skin-color extraction, correlation-based matching, and multiple-scale image generation from a single camera. These auditory and visual streams are associated by comparing the spatial location, and associated streams are used to control focus of attention. Auditory, visual, and association processing are performed asynchronously on different PC's connected by TCP/IP network. The resulting system implemented on an upper-torso humanoid can track multiple objects with the delay of 200 msec, which is forced by visual tracking and network latency.

1 Introduction

Humanoids and entertainment robots or at least mobile robots have attracted a lot of attention last year, e.g., at IEEE/RSJ First Humanoids-2000 conference, and are expected to play a role of human partners in the 21st century. Let us imagine the situation autonomous robots are used in social and home environment, such as a pet robot at living room, a service robot for office, or a robot serving people at a party. The robot shall identify people in the room, pay attention to their voice and look at them to identify visually, and associate voice and visual images, so that highly robust event identification can be accomplished. These are minimum requirements for social interaction [Brooks *et al.*, 1998].

Some robots are equipped with improved robot-human interface. *Jijo-2* [Asoh *et al.*, 1997] can recognize a phrase command by speech-recognition system; *AMELLA* [Waldherr *et al.*, 1998] can recognize pose and motion gestures. *Kismet* of MIT AI Lab [Breazeal and Scassellati, 1999] can recognize speeches by speech-recognition system and express various kinds of sensation. *Hadaly* of Waseda University [Matsusaka *et al.*, 1999] can localize the speaker as well as recognize speeches by speech-recognition system.

However, the technologies developed so far are still immature; in particular, auditory processing and integrated perception among vision, audition, and motor control. At robotic conferences such as IROS, SMC, and ICRA as well as AI-related conferences, there were at most one or two papers related to auditory processing[1], and most papers on robot perception is limited to vision-only and vision with ultrasonic, infra-red or laser range finders. This is unfortunate because integrated processing of auditory and visual processing combined with appropriate motor control is essential in social interaction of robot systems.

For auditory and visual tracking, Nakadai *et al.* presented the *active audition* for humanoids to improve sound source tracking by integrating audition, vision, and motor controls [Nakadai *et al.*, 2000]. An active audition system is implemented in a upper-torso humanoid to demonstrate that the humanoid actively moves its head to improve localization by aligning microphones orthogonal to the sound source and by capturing the possible sound sources by vision. Although such an active head movement inevitably creates motor noise, the system adaptively cancels motor noise using motor control signals. The experimental result demonstrates that the active audition by integration of audition, vision, and motor control enables sound source tracking in variety of conditions. One of crucial problems is a lack of real-time processing.

Matsuyama *et al.* presented an architecture for asynchronous coordination of sensorimotor control [Matsuyama *et al.*, 2000] so that the camera moves smoothly to track the

[1] Auditory processing shall be distinguished from speech recognition. Auditory processing (auditory scene analysis) aims at separation and understanding of multiple sound streams, such as human speech, environmental noise, music, etc.

object in real-time. This architecture, called *dynamic memory*, is general, but they use it only for vision-based motor control. Shafer *et al.* presented the software architecture of sensor fusion for an autonomous mobile robot [Shafer *et al.*, 1986]. The architecture is based on a parallel blackboard system, and the sensors include vision, range finder, but not microphones. It exploits global consistency regarding position and orientation of the vehicle and sensors. Murphy presented the sensor fusion system for mobile robots called the Sensor Fusion Effects (SFX) architecture, which is based on the uncertainty management system by Dempster-Shafer theory [Murphy, 1998].

Other robots with microphones as ears for sound source localization or sound source separation have attained little in auditory tracking. *Kismet* has a pair of omni-directional microphones outside the simplified pinnae [Breazeal and Scassellati, 1999]. Since it is designed for one-to-one communication and its research focuses on social interaction based on visual attention, the auditory tracking has not been implemented so far. *Hadaly* uses a microphone array to perform sound source localization, but the microphone array is mounted in the body and its absolute position is fixed during head movements [Matsusaka *et al.*, 1999]. In the both cases, sound source separation is not exploited and a microphone for speech recognition is attached to the speaker.

In the research of computational auditory scene analysis (CASA) to understand a mixture of sounds, real-time processing is one of the main problems in applying CASA to real-world applications [Rosenthal and Okuno, 1998]. Real-time processing is important to take appropriate actions in daily environments where many people, robots and objects exist. Auditory and visual tracking by Nakadai *et al.* accepts sounds in daily environment, but does not run in real-time [Nakadai *et al.*, 2000]. Another auditory and visual tracking by Nakagawa *et al.* does not run in real-time [Nakagawa *et al.*, 1999].

Two major issues that have not been done in the past are attacked in this paper, that is, association of multiple auditory and visual streams, and real-time processing of integrated auditory and visual scene analysis.

The rest of the paper is organized as follows: Section 2 explains the robot hardware which is used as a testbed and presents the issues in real-time tracking. Section 3 describes the design of the system and the details of each module are described in Section 4. Section 5 demonstrates and evaluates the performance of the system. Section 6 discusses the observations of the experiments and future work and concludes the paper.

2 Issues in Real-Time Tracking

2.1 Robot Hardware

As a testbed of real-time multiple-object tracking, we use a upper-torso humanoid called *SIG* shown in Fig. 1 [Nakadai *et al.*, 2000]. The cover of the mechanics is made of FRP and discriminates internal and external world acoustically. *SIG* has two microphones at the left and right ear positions to capture external sounds from outside of the body, and two microphones within the body to capture internal sounds mainly caused by motor movements. All the microphones are omni-directional microphones of Sony ECM-77S. *SIG*'s body has four DOFs (degree of freedom), each of which is a DC motor controlled by a potentiometer. *SIG* is equipped with a pair of CCD cameras of Sony EVI-G20, but the current vision module uses only one camera.

Figure 1: Humanoid, *SIG*

2.2 Task and Issues

The task in this paper is to track multiple objects in real-time with two kinds of sensors, a camera and two microphones, and one actuator to rotate the body. Some important issues in this task are:

- Robustness in visual stream extraction (temporal sequences of face localization and face identification) against non-uniform environments due to lighting conditions or moving humans.
- Robustness in auditory stream extraction (temporal sequence of sound source localization and sound source separation) against dynamic environments because objects (people) move and the humanoid also moves.
- Association of visual and auditory streams by face identification and sound source localization to compensate missing or ambiguous data. Common representation for both auditory and visual feature extractions is needed.
- Focus of attention control based on association to control actuators.
- Trade-off of processing speed vs quality of feature extractions for real-time processing.
- Method of synchronization between asynchronous auditory and visual processing that have different processing speeds. The frame rate of vision is 30 Hz, while the sampling rate of sound is 48 KHz. Therefore, asynchronous processing is essential to exploit the full range of concurrency.

3 Design of the System

From the viewpoint of functionality, the whole system can be decomposed into five layers — *SIG* Device Layer, Process Layer, Feature Layer, Event Layer, and Stream Layer

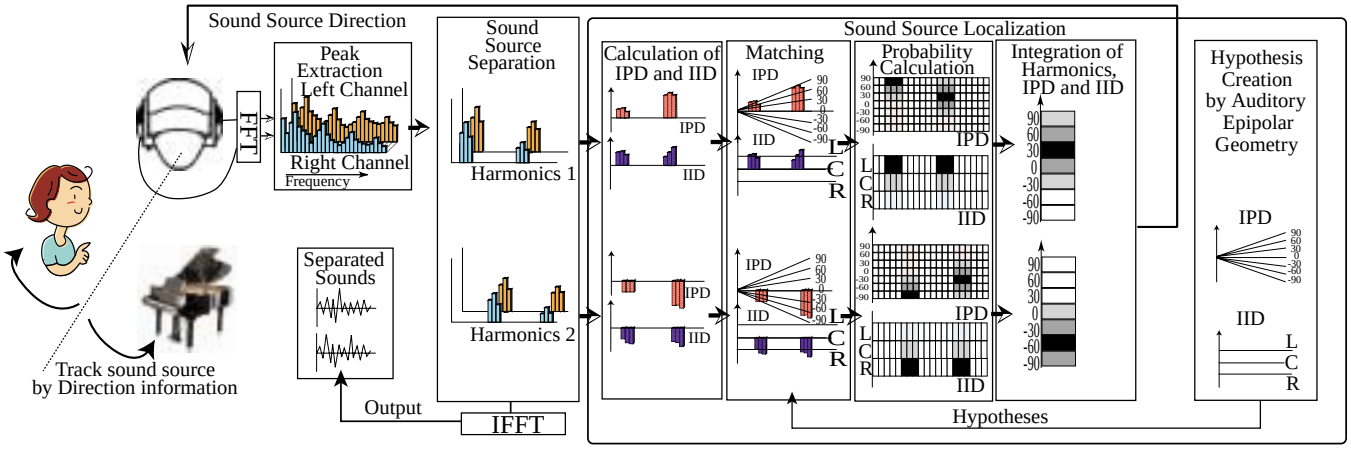


Figure 3: Audition Module extracts harmonic structures to localize and separate sound sources.

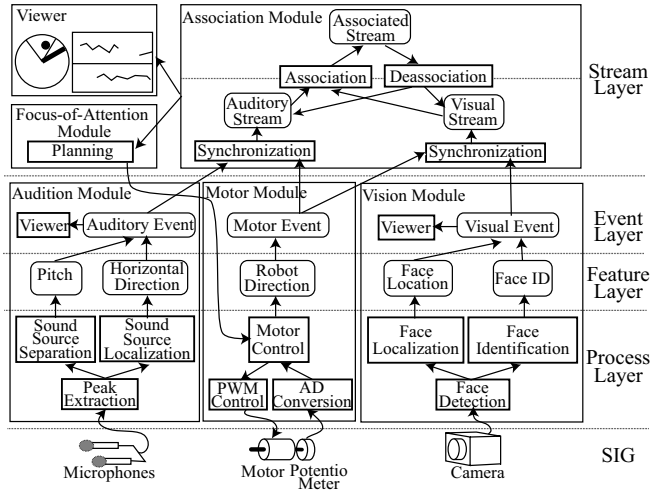


Figure 2: Modules and Layers of the System

(see Fig. 3) . From the viewpoint of implementation, the whole system consists of six asynchronous modules — Audition, Vision, Association, Focus-of-Attention, Motor Control, and Viewer. This relation between two viewpoints is depicted in Fig. 2. Since Vision module utilizes the full power of Pentium-III 733 MHz CPU, the whole system is organized in the form of distributed processing with three Linux nodes based on Pentium-III with RedHat Linux 6.2J. The first node with 600 MHz is for Audition, the second node with 733 MHz for Vision, and the third with 450 MHz for the rest. They are connected by TCP/IP over Fast Ethernet 100Base-TX.

The estimated processing time of each module executed on a node is summarized below:

- Vision – 200 msec for face localization and identification,
- Audition — 40 msec for sound source localization,
- Motor Control — 100 msec
- Network latency — up to 200 msec

Therefore, we set the goal that the response time of the system should be 200 msec of delay.

Audition and Vision generate an *event* by feature extraction and organize a *stream* as a temporal sequence of events. Motor Control also generates an event of motion. Association fuses these events to make a higher level representation. This fusion *associates* auditory and visual streams to make an *associated* stream. Focus-of-Attention makes a planning of *SIG*'s movement based on the status of streams, that is, whether they are associated or not.

Motor Control is activated by Focus-of-Attention module and generates PWM (Pulse Width Modulation) signals to DC motors. It also sends a motor event consisting of motor direction (azimuth of the midsagittal plain) to Association module. Viewer shows the status of auditory, visual and associated streams in the radar and scrolling windows (see screen shots shown in Fig. 5). Some modules are explained in details in the next section.

4 Details of Each Module

4.1 Active Audition Module

Sound localization for a robot or an embedded system is usually solved by using interaural phase difference (IPD) and interaural intensity difference (IID). These values are calculated by using Head-Related Transfer Function (HRTF). However, HRTF depends on the shape of head and it also changes as environments change. For real-world applications, sound localization without HRTF is preferable. Nakadai *et al.* proposed the method based on the auditory epipolar geometry, an extension of epipolar geometry in stereo vision to audition [Nakadai *et al.*, 2000]. They also proposed *active audition* for sensorimotor task with canceling motor and mechanical noises. However, they failed in doing the jobs in real-time, because they stuck to pure-tone processing. In this paper, we extend their approach (1) by exploiting the harmonic structure to extract peaks precisely and (2) by solving the uncertainty in sound source localization by Dempster-Shafer theory.

Audition module equipped with active audition is depicted in Fig. 3. The input signal, a mixture of sounds originating from different directions, is sampled with sampling frequency

Table 1: Belief Factor of IID, $BF_{\mathrm{IDD}}(\theta)$

	θ	$90^\circ \sim 35^\circ$	$30^\circ \sim -30^\circ$	$-35^\circ \sim -90^\circ$
I	+	0.35	0.5	0.65
	-	0.65	0.5	0.35

of 48 KHz and 16-bit quantization, and its spectrogram is calculated by Fast Fourier Transforms (FFT). Audition extracts pitches (fundamental frequency, $F0$), separates and localizes sound sources.

Peak Extraction and Sound Source Separation: First a peak is extracted by a band-pass filter, which passes a frequency between 90 Hz and 3 KHz if its power is a local maximum and more than the threshold. This threshold is automatically determined by the stable auditory conditions of the room. Then, extracted peaks are clustered according to *harmonicity*. A frequency of Fn is grouped as an overtone (integer multiple) of $F0$ if the relation $\left|\frac{Fn}{F0} - \lfloor\frac{Fn}{F0}\rfloor\right| \leq 0.06$ holds. The constant, 0.06, is determined by trial and error. By applying Inverse FFT to a set of peaks in harmonicity, a harmonic sound is separated from a mixture of sounds.

Sound Source Localization: Once a harmonic structure is obtained, the direction of sound source is calculated by hypothetical reasoning for IPD (Interaural Phase Difference) and IID (Interaural Intensity Difference). The azimuth (horizontal direction) is quantized and represented by every 5° discrete value in the range of $\pm 90^\circ$. The front direction of *SIG* is 0°.

From the extracted harmonic structure of left and right channels, a pair of harmonic structures is obtained. Then the IPD, P_s, is calculated. Auditory Epipolar Geometry generates a hypothesis of IPD P_h for each 5° candidate, θ [Nakadai *et al.*, 2001]. Since the IPD is ambiguous for frequencies of more than 1200 Hz, the distance, $d(\theta)$, in IPD between the data and a hypothesis is defined as follows:

$$d(\theta) = \frac{1}{n_{f<1200\mathrm{Hz}}} \sum_{f=F0}^{1200\mathrm{Hz}} \frac{\left(P_h(\theta, f) - P_s(f)\right)^2}{f} \qquad (1)$$

Where $n_{f<1200Hz}$ is the number of overtones of which frequency is less than 1200 Hz.

The similar relation may hold for IID, but our experience with IID proves that it can discriminate at most the side, that is, left or right. Suppose that $I_s(f)$ is the IID for peak frequency f. If the value of $I = \sum_{f=1200\mathrm{Hz}}^{3000\mathrm{Hz}} I_s(f)$ is non-negative, the direction is decides as left, otherwise as right:

Integration of IPD and IID by Dempster-Shafer theory To determine the sound source direction, the belief factors of IPD and IID are calculated and then integrated by Dempster-Shafer theory. The belief factor of IPD, BF_{IPD}, is calculated by using probability density function defined by Eq. (2).

$$BF_{\mathrm{IPD}}(\theta) = \int_{-\infty}^{\frac{d(\theta)-m}{\sqrt{\frac{s}{n}}}} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx \qquad (2)$$

where m and s are the average and variance of $d(\theta)$, respectively. n is the number of d.

The belief factor of IID, $BF_{\mathrm{IID}}(\theta)$ is defined by Table 1.

Then, belief factors of IPD and IID, BF_{IPD} and BF_{IID}, are integrated using Dempster-Shafer theory as defined in Eq. (3).

$$BF_{\mathrm{IPD+IID}}(\theta) = BF_{\mathrm{IPD}}(\theta)BF_{\mathrm{IID}}(\theta) + \left(1 - BF_{\mathrm{IPD}}(\theta)\right)BF_{\mathrm{IID}}(\theta) + BF_{\mathrm{IPD}}(\theta)\left(1 - BF_{\mathrm{IID}}(\theta)\right) \qquad (3)$$

θ for the maximum $BF_{\mathrm{IPD+IID}}$ is treated as the sound source direction of the harmonics. Finally, Audition sends an auditory event consisting of pitch ($F0$) and a list of 20-best directions (θ) with reliability factor for each harmonics.

4.2 Real-Time Multiple Face Tracking

Multiple face detection and identification suffers more severely from frequent changes in the size, direction and brightness of face. To cope with this problem, Hidai *et al.* combines skin-color extraction, correlation based matching, and multiple scale images generation [Hidai *et al.*, 2000].

The requirements on multiple face tracking are the capability of discriminating face data of the same face ID from others and on-line learning. The first requirement is a class concept. Turk *et al.* proposed the eigenface matching technique as a kind of subspace method [Turk and Pentland, 1991]. A subspace for discrimination is created by Principal Component Analysis (PCA). PCA, however, does not provide the means to group a data according to its face ID, since such an ID cannot be generated by PCA. Thus, the subspace obtained by PCA is not always suitable to distinguish such classes.

On the other hand, Liner Discriminant Analysis (LDA) can create an optimal subspace to distinguish classes. Therefore, we use Online LDA [Hiraoka *et al.*, 2000]. In addition, this method continuously updates a subspace on demand with a small amount of computation.

The face identification module (see Fig. 2) projects each extracted face into the discrimination space, and calculates its distance d to each registered face. Since this distance depends on the degree (L, the number of registered faces) of discrimination space, it is converted to a parameter-independent probability P_v as follows.

$$P_v = \Gamma\left(\frac{1}{2}, \frac{d^2}{2}\right) = \int_{\frac{d^2}{2}}^{\infty} e^{-t}\, t^{\frac{L}{2}-1} dt \qquad (4)$$

The face localization module converts a face position in 2-D image plane into 3-D world coordinate. Suppose that a face is $w \times w$ pixels located in (x, y) in the image plane, whose width and height are X and Y, respectively (see screen shots shown in Fig. 5). Then the face position in the world coordinate is obtained in terms of distance r, azimuth θ and elevation ϕ by the following equations.

$$r = \frac{C_1}{w},\ \theta = \sin^{-1}\left(\frac{x - \frac{X}{2}}{C_2\, r}\right),\ \phi = \sin^{-1}\left(\frac{\frac{Y}{2} - y}{C_2\, r}\right)$$

where C_1 and C_2 are constants defined by the size of the image plane and the image angle of the camera.

Finally, Vision module sends a visual event consisting of a list of 5-best Face ID (Name) with its reliability and position (distance r, azimuth θ and elevation ϕ) for each face.

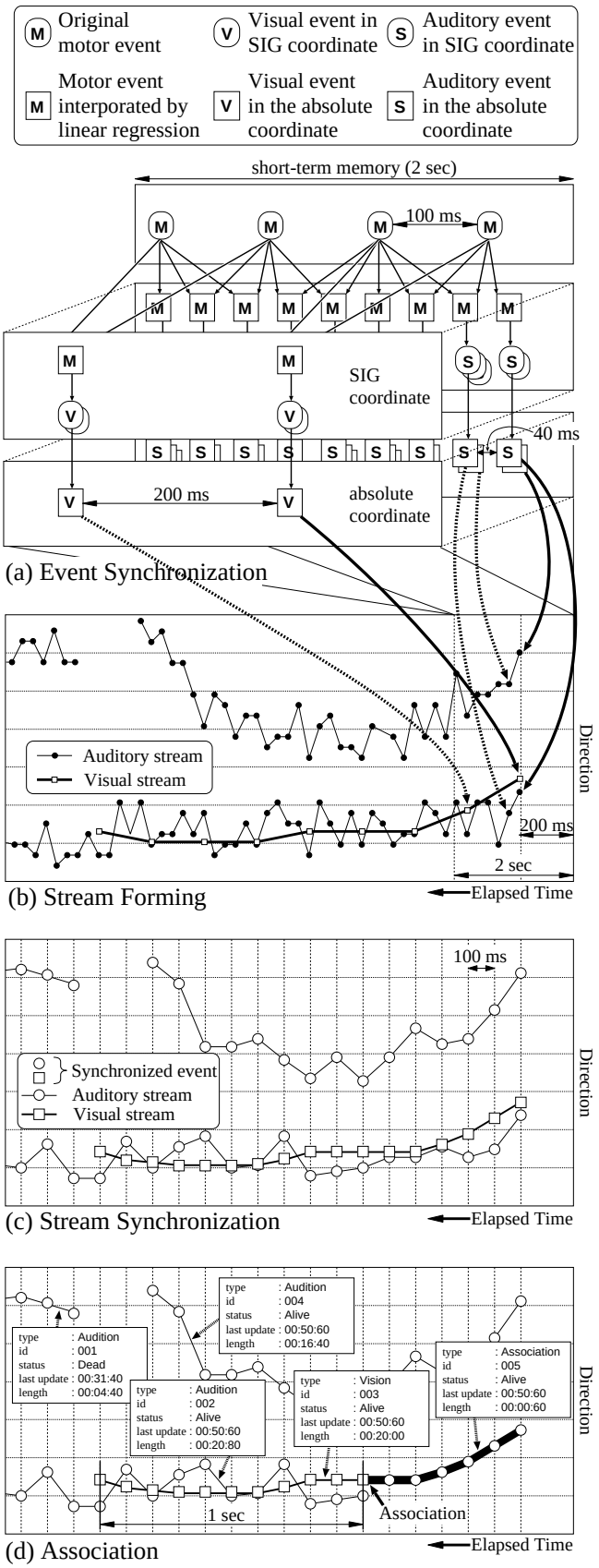


Figure 4: Association Module Forms Streams

4.3 Stream Formation and Association

Association module forms auditory streams from auditory events and visual streams from visual events, and associates a pair of auditory and visual streams to create a higher level stream, which is called an *associated stream* (see Fig. 2). The flow of processing in stream formation and association is summarized as follows (Figs. 4(a)-(d)):

1. Events from Audition, Vision and Motor modules are stored in the short-term memory.
2. Direction information of events is converted into the absolute coordinate to treat them in the common coordinate.
3. Events are grouped into an auditory or a visual stream according to a temporal sequence of events.
4. Streams are synchronized by every 100 msec to calculate the distance between streams.
5. An auditory and a visual stream which are close for more than a constant time are associated as an associated stream.

First, events are stored in the short-term memory and kept only for 2 seconds to attain incremental and real-time processing. In Fig. 4(a), where Ⓢ, Ⓥ and Ⓜ represent events created by Audition, Vision and Motor modules, respectively. Each module creates events at its own cycle, e.g. 40 msec for audition, 200 msec for vision and 100 msec for motion. Then, motor events are synchronized with auditory and visual events. To put it concretely, a motor direction when an auditory or a visual event appeared is estimated from motor events in the short-term memory. A motor event with the estimated motor direction is shown as [M]. Because visual events from Vision module and auditory events from Audition module are represented in robot coordinate, the directions of these events are converted to ones in the absolute coordinate by using estimated motor direction. These are represented as [S] or [V] in Fig. 4(a). This synchronization process runs with a delay of 200 msec as mentioned in Sec. 3.

Auditory and visual streams are formed in Fig. 4(b). X-axis indicates elapsed time from right to left, and Y-axis indicates azimuth in the absolute coordinate. Thin lines with small filled circles and a thick line with small rectangles represent auditory streams and a single visual stream. An auditory event is connected to the nearest auditory stream within the range of $\pm 10^\circ$ and with common $F0$. A visual event is connected to the nearest visual stream within 40 cm and with a common face ID. In either case, if there are multiple candidates, the most reliable one is selected. If any appropriate stream is found, such an event becomes a new stream. In case that no event is connected to an existing stream, such a stream remains alive for up to 500 msec. The system cannot detect an auditory or a visual event when a person stops talking and looks away for a moment. This margin of 500 msec is prepared to continue streams in case of the missing event extraction. After 500 msec of keep-alive state, the stream terminates.

When the distance between an auditory and a visual stream is close for more than a constant time, they are regarded as

streams originating from the same object and integrated into an associated stream, which is a higher layer representation of a stream shown in Fig. 2. Because auditory and visual streams consist of events with 40 msec and 200 msec cycles, respectively, it is difficult to evaluate the distance between these two streams without synchronization. Then, they are synchronized with the same cycle, 100 msec. Fig. 4(c) illustrates synchronized streams as lines with large circles and rectangles. If an event is not available in this case, linear regression is used for interpolation in the same way as synchronization with motor events.

An auditory and a visual streams are associated if their direction difference is within the range of $\pm 10^\circ$ and this situation continues for more than 50% of the 1 sec period shown in Fig. 4(d).

The visual direction is usually used for the direction of the associated stream because visual information is more accurate. However, when a tracking person is occluded, the system cannot use visual information. In this case, auditory information is used for the associated stream. This suggests an advantage of integration of audition and vision, i.e. auditory information is efficient not only for pre-attentive uses such as a trigger of attention but also for compensations of missing or ambiguous information as this case. If either auditory or visual event has not been found for more than 3 sec, such an associated stream is deassociated and only existing auditory or visual stream remains. If the auditory and visual direction difference has been more than 30° for 3 sec, such an associated stream is deassociated to two separate streams.

4.4 Focus of Attention Control

SIG should pay attention for sounds from unseen objects to get further information. When such a sound does not exist, faces with sound, i.e. talking people, should have high priority because they are attractive even for human perception. The principle of focus-of-attention control hereby is as follows:

1. An auditory stream has the highest priority,
2. an associated stream has the second priority, and
3. a visual stream has the third priority.

The algorithm of focus-of-attention control is sketched by using an example shown in Fig. 5, which depicts how auditory and visual streams are generated and associated.

1. Focus of attention changes to a new association stream. [t_1 and t_8 of Fig. 5].
2. If one of the visual and auditory stream of an associated stream terminates due to occlusion, disappearance, or end of speech, association continues [t_4 to t_5 of Fig. 5].
3. If this state continues for a particular time, say 3 seconds, the focus of attention may change.
   (a) Focus of attention changes to one of associated streams.
   (b) If no associated stream is found, focus of attention changes to one of auditory streams. [t_6 of Fig. 5].
   (c) Otherwise, focus of attention changes to one of visual streams.
4. In turning the body to associate the auditory stream to visual one, focus of attention keeps the same even if a new associated stream is generated.

5 Experiments and Evaluation

A 40-second scenario shown in Fig. 5 is used as a benchmark. The performance of integrated auditory and visual tracking is shown in Fig. 5, which shows that focus of attention changes twice. In the first half of the scenario up to $t = 26$ sec, two speakers are apart, while in the second half they are close and viewed in the same camera view field. In both cases, the system can track the speakers well.

The direction of *SIG*'s body is depicted in Fig. 6, which shows that the motor control succeeds in giving correct PWM motor commands. To sum up, sensorimotor task in single- and multi-speaker tracking is well accomplished.

The performance of visual tracking is shown in Fig. 7. This timechart is generated by collecting the first candidate from the internal states of Vision module. Therefore, the motor movement is the same as the above. In the first half of the scenario, occlusion causes a gap of visual streams between t_4 and t_5. From t_6 to t_7, no person can be seen due to the limited range of camera view field. Fig. 5 proves that occlusion and out-of-sight can be easily recovered by associated streams.

The performance of auditory tracking is shown in Fig. 8, which is generated in the same manner as Fig. 7. The Audition module can separate two auditory streams correctly from t_3 to $t = 23$ sec, and t_9 to t_{10}, but generate erroneous streams around t_8 and t_9. In addition, the directions of two speakers are not so correct from $t = 11$ sec (t_5) to $t = 17$ sec, because Mr. A moves and *SIG* tracks him by rotating its body. That is, the reverberation (echo) due to this moving talker and motor noise deteriorates the quality of sound source localization.

5.1 Limitations on the Proposed System

The room used in this experiment is about $3m$ in width and length and $2m$ in height, and sound absorbing materials are attached on walls, ceiling and floor. It is not anechoic, but has reverberation time of 0.1 sec. Because the value in a normal speech studio of the equivalent size is about 0.2 sec, the room has less reverberation. Acoustic conditions, however, depends on objects in the room. When we put a plastic partition of 1.5 m $\times$ 1.5 m with strong reverberation in the room, the correctness of sound source localization is reduced remarkably.

The background noise level of the room is 30 dBA on average. This is measured with the filter by *A weighting*, similar to human auditory characteristic. The value of 30 dBA corresponds to the noise level of a room in a quiet residence. We confirmed that the current system works well up to 40 dBA of background noise level, but we have not checked above 40 dBA.

The room has six halogen lights with adjustment function of the intensity on the ceiling. The range of the light intensity is from 0 to 50 lux. Because the light intensity from 300 to 1500 lux is recommended for a normal office by JIS (Japanese Industrial Standard), even the maximum light intensity of the room is weak. Our face extraction and recognition method works well under the condition of more than 5 lux. Visual

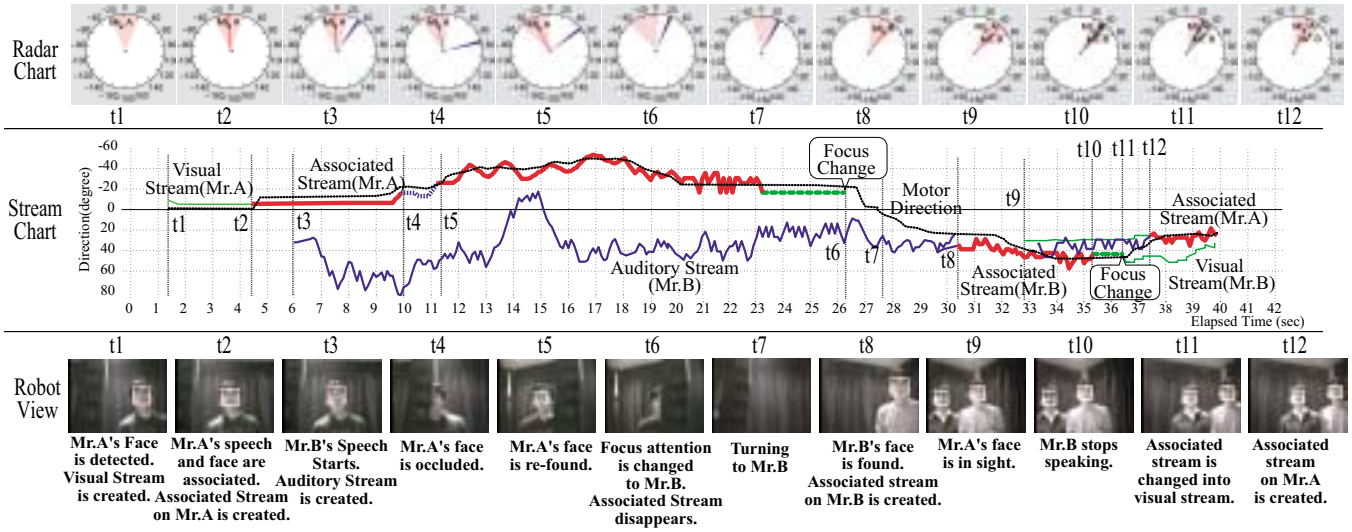


Figure 5: Temporal Sequence of Auditory and Visual Tracking of Two Speakers: **Radar Chart** and **Stream Chart** are screen shots of the viewer. In radar chart, a wide-light and a narrow-dark sector indicate the camera view field and sound source direction, respectively. In stream chart, a thin line indicates auditory or visual stream, while a thick line indicates an associated stream.

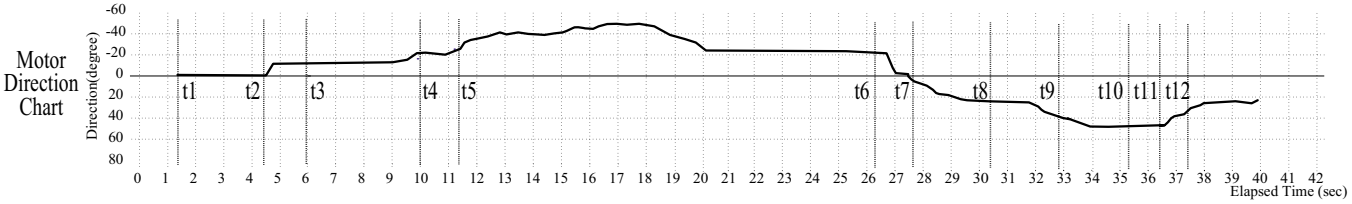


Figure 6: Temporal Sequence of Body Direction controlled by Motor Movement in the same scenario as Fig. 5

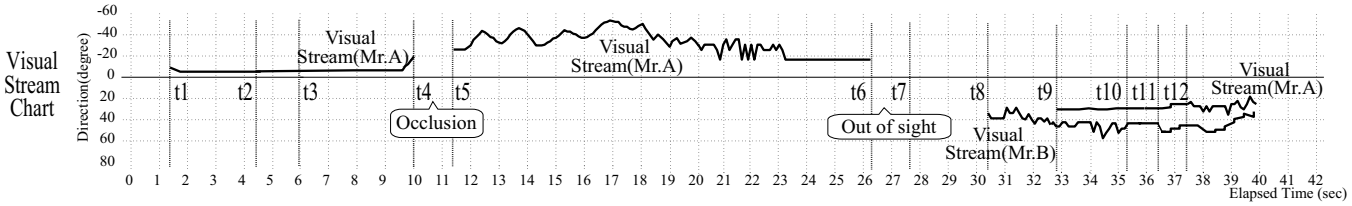


Figure 7: Temporal Sequence of Visual Tracking of Two Speakers in the same scenario as Fig. 5

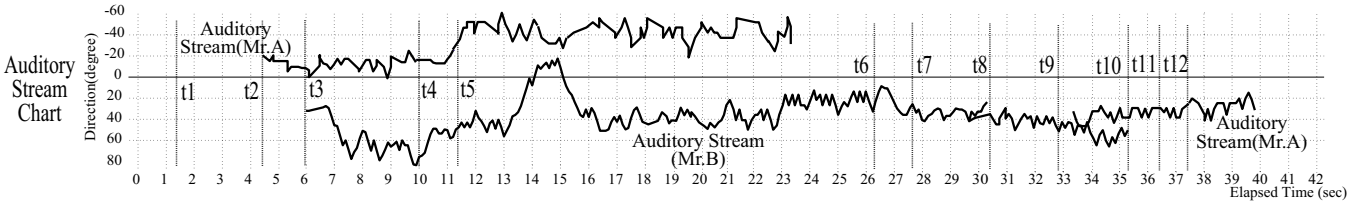


Figure 8: Temporal Sequence of Auditory Tracking of Two Speakers in the same scenario as Fig. 5

processing is robust against the change of light intensity. In this experiment, the light intensity in the room is 50 lux.

Other benchmarks such as crossing of moving talkers, moving talkers without seeing any talkers, and alternative talking of four speakers prove that the resulting system succeeds in real-time sensorimotor tasks of tracking [2].

[2]Since the work is related to the real-time processing, the readers may be suggested to visit the following Web site: http://www.symbio.jst.go.jp/SIG/

6 Conclusion and Future Work

The key idea of real-time tracking is "*For each processing, take it easy, and ambiguities will be resolved with the help of others.*" This idea is obtained by the scrutiny of the behavior of each component of implementations of Nakadai *et al.*'s work [Nakadai *et al.*, 2000]. We do not stick to pure tones, but utilize the collective behavior of harmonic sounds; we prefer frequency resolution over the time resolution by increasing the points of FFT. We give up the precise face localization and identification. Instead, we associate auditory, visual, and motor direction information to localize the sound sources.

Some technical future work includes learning the adaptive association of different or dynamic environments. Since lighting conditions and reverberation (echo) change drastically in such environments, Vision and Audition modules should adjust their parameters on demand. In addition, Association should adapt its parameters for stream forming and association. Baysian algorithm for resolving ambiguities in stream forming and association is a promising technique. Another future work is incorporating stereo vision. Even in a static environment robust auditory processing such as sound source separation and localization would be useful when a room is noisier, has objects with strong reverberation, or has many people.

We believe that our result would open a new era of sound processing, in particular, cocktail party computer or "Shotoku-Taishi" computer that can listen to several things at once.

Auditory and visual tracking should be incorporated in a total system with robot-human interface. We have already built such a system comprising speech recognition, speaker identification, and speech synthesis based on the proposed system. Once the application is fixed, the top-down stream separation may be exploited. Some information that forces top-down stream separation includes speaker identification. The speaker information may reduce the search space of face recognition and speech recognition. For example, let us consider that crossing of two talking persons. In this case, the system may miss judging that they are approaching and then receding because speaker IDs are lacking. Thus, speaker identification can reduce this kind of ambiguity. Its design and implementation will be reported by a separate paper, since this paper focuses on the real-time processing.

Acknowledgments

We thank our colleagues of Symbiotic Intelligence Group, Kitano Symbiotic Systems Project; Mr. Tatsuya Matsui and Dr. Tino Lourens, and Prof. H. Ishiguro of Wakayama University for their discussions.

References

[Asoh *et al.*, 1997] H. Asoh, S. Hayamizu, I. Hara, Y. Motomura, S. Akaho, and T. Matsui. Socially embedded learning of the office-conversant mobile robot *jijo-2*. In *Proceedings of 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, volume 1, pages 880–885. AAAI, 1997.

[Breazeal and Scassellati, 1999] C. Breazeal and B. Scassellati. A context-dependent attention system for a social robot. In *Proceedints of the Sixteenth International Joint Conference on Atificial Intelligence (IJCAI-99)*, pages 1146–1151, 1999.

[Brooks *et al.*, 1998] R. A. Brooks, C. Breazeal, R. Irie, C. C. Kemp, M. Marjanovic, B. Scassellati, and M. M. Williamson. Alternative essences of intelligence. In *Proceedings of 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 961–968. AAAI, 1998.

[Hidai *et al.*, 2000] K. Hidai, H. Mizoguchi, K. Hiraoka, M. Tanaka, T. Shigehara, and T. Mishima. Robust face detection against brightness fluctuation and size variation. In *Proceedings of IEEE/RAS International Conference on Intelligent Robots and Systems (IROS-2000)*, pages 1397–1384. IEEE, 2000.

[Hiraoka *et al.*, 2000] K. Hiraoka, S. Yoshizawa, K. Hidai, M. Hamahira, H. Mizoguchi, and T. Mishima. Convergence analysis of online linear discriminant analysis. In *Proceedings of IEEE-INNS-ENNS International Joint Conference on Neural Networks*, volume III, pages 387–391, 2000.

[Matsusaka *et al.*, 1999] Y. Matsusaka, T. Tojo, S. Kuota, K. Furukawa, D. Tamiya, K. Hayata, Y. Nakano, and T. Kobayashi. Multi-person conversation via multi-modal interface — a robot who communicates with multi-user. In *Proceedings of 6th European Conference on Speech Communication Technology (EUROSPEECH-99)*, pages 1723–1726. ESCA, 1999.

[Matsuyama *et al.*, 2000] T. Matsuyama, S. Hiura, T. Wada, K. Murase, and A. Yoshida. Dynamic memory: Architecture for real time integration of visual perception, camera action, and network communication. In *Proceedings of ICCV*, pages 728–735. IEEE, 2000.

[Murphy, 1998] R.R. Murphy. Dempster-shafer theory for sensor fusion in autonomous mobile robots. *IEEE Transactions on Robotics and Automation*, 14(2):197–206, 1998.

[Nakadai *et al.*, 2000] K. Nakadai, T. Lourens, H. G. Okuno, and H. Kitano. Active audition for humanoid. In *Proceedings of 17th National Conference on Artificial Intelligence (AAAI-2000)*, pages 832–839. AAAI, 2000.

[Nakadai *et al.*, 2001] K. Nakadai, H. G. Okuno, and H. Kitano. Epipolar geometry based sound localization and extraction for humanoid audition. *submitted*, 2001.

[Nakagawa *et al.*, 1999] Y. Nakagawa, H. G. Okuno, and H. Kitano. Using vision to improve sound source separation. In *Proceedings of 16th National Conference on Artificial Intelligence (AAAI-99)*, pages 768–775. AAAI, 1999.

[Rosenthal and Okuno, 1998] D. Rosenthal and H. G. Okuno, editors. *Computational Auditory Scene Analysis*. Lawrence Erlbaum Associates, Mahwah, New Jersey, 1998.

[Shafer *et al.*, 1986] S.A. Shafer, A. Stentz, and C.E. Thorpe. An architecture for sensor fusion in a mobile robot. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 2002–2011. IEEE, 1986.

[Turk and Pentland, 1991] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[Waldherr *et al.*, 1998] S. Waldherr, S. Thrun, R. Romero, and D. Margaritis. Template-based recoginition of pose and motion gestures on a mobile robot. In *Proceedings of 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 977–982. AAAI, 1998.