

Routine Based Models of Anticipation in Natural Behaviors

Weilie Yi and Dana H. Ballard

Department of Computer Science
University of Rochester
Rochester, NY 14627
{wyi, dana}@cs.rochester.edu

Abstract

This paper describes the analysis and modeling of natural human behaviors on top of a routine based model of cognition. This theory hypothesizes that there exists in the brain a collection of built-in programs, which are coded with a fixed set of basic visual primitives and can be reprogrammed to carry out various visual tasks. Despite the general recognition of the value of such an approach and several pieces of biological evidence supporting it, no detailed successful model of visual routines has been built. Furthermore, no situated model has been described that acknowledges the specific advantages conferred on such an approach by the human body and elaborate eye-movement system. We built a real-time computer vision system that can simulate basic human visuo-motor behaviors, and provided a platform which facilitates the study of human world interaction as sequential assembly of perceptual primitives. Based on this platform, we break down natural behaviors by automatic subtask recognition, and model task planning with a Markov model. The Markov model effectively allows subjects to anticipate by capturing the statistics of previous trials.

Introduction

During the past years, researchers in psychology and physiology have generally reached a consensus that human vision system is an active vision system (Ballard 1991) (Findlay & Gilchrist 2003). In the active vision paradigm, vision system is actively *seeking* for information that is relevant to current cognitive activity. The non-uniform spatial distribution of photoreceptors in the retina is one observation point in a complex chain of reasoning that suggests the vision system cannot, and does not have to, reconstruct the 3D model of the scene in the brain, but rather samples pieces of information from the scene by initiating eye movements that bring the object of interest into the fovea. Because the high resolution retinal image being processed is extremely local in the fovea (Yarbus 1967), the process of any non-trivial visual action is a temporal sequence of both overt shifts of the fixation point (Noton & Stark 1971) and the movements of covert focus. Thus it is natural to model visual actions as sequences of small steps.

Copyright © 2005, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

Imagine we can make high-speed snapshots of a segment of visual behavior, including the physical movement of the eyes, shifting of internal focus, memory manipulation, and other changes of internal states. If we only keep those snapshots where important events happen, e.g. an object is recognized and remembered, or a position is fixated on, we can describe the visual behavior as a sequence of such events. If we further hypothesize that there is a finite, or even small number of these events, or basic operations, we have the basis for the *visual routines theory*. This theory (Ullman 1996) claims that there is a repository of precompiled programs in the brain, each of which could be triggered on demand when a relevant task is present, and, these programs are assembled from a fixed set of primitive operations. These programs, called visual routines, are assumed to be pre-wired into the brain's visual system and can be created and modified on demand.

A routine based cognitive system can take advantage of the interactions with its surroundings, exploiting embodied information through the use of eyes, hands and other body parts. This viewpoint of embodied cognition emphasizes that cognitive processes are deeply rooted in the body's real time interactions with the world, which inherently involves perception and action, and cognitive mechanisms "must be understood in terms of their ultimate contribution to situation-appropriate behavior" (Wilson 2002). Due to the limits on attention and working memory, humans exploit the environment to reduce the cognitive workload. We make the task dependent environment store or even compute information for us, and we retrieve that information only on a need-to-know basis.

Since the visual routine model was proposed, researchers in both AI community and cognitive psychology community have been trying to implement it. One of the earliest such implementations was made by Chapman (Chapman 1991), who designed an autonomous agent which can play a 2-D video game. About 40 basic operations, which are highly task specific, formed the instruction set. Johnson (Johnson, Maes, & Darrell 1994) used evolutionary computation to automatically generate visual routines to do image classification. In Horswill's *Jeeves* system (Horswill 1995), that was claimed to be the first real time implementation of visual routine on real world data, various color maps, intensity, temporal and spatial derivatives, and even optic flow were

calculated in early vision stage. Kahn and Swain(Kahn & Swain 1995) were trying to mimicked the early vision in human visual system by calculating intensity, edge, color, motion and disparity in their Perseus system whose job was gesture recognition. Salgian(Salgian & Ballard 1998) developed an autonomous driving system with three visual routines, stop light detection, stop sign detection and car following. Beetz et al.(Beetz *et al.* 1998) built an autonomous robot in which image processing are combined with robot control in a high level language. Although all of this research is either directly or indirectly based on the visual routine model, none provides a general-purpose framework for both visual routines, embodied active vision and motor action.

This paper introduces a routine based computer vision architecture with a basic operation set which is task independent. The active vision approach is taken, and both visual and motor movements form the backbone of the interactions with the environment. This system concentrates on high level visual cognition and bypasses low level vision such as image segmentation and object recognition, and assumes these processes are available to use. Because of the marriage between visual routines and active vision, it avoids reconstructing the 3D model of the world. Geometric information is retrieved on demand by executing proper routines, so the system can perform tasks in a geometry independent fashion. The system simulates human visuo-motor cognition in real time in a 3D virtual environment. The reason we do not use hardware robots is that virtual agents are more cost effective, flexible, extendable and allow for faster algorithm development.

A key of the system is the use of a Markov model to capture interpersonal differences in solving natural tasks. A task is automatically segmented into subtasks and the execution order is determined in a probabilistic fashion. Even if not fully unrolled, a plan exists before the task begins. In a very real sense the Markov model exhibits the ability to *anticipate* events because 'the past is prologue.' Having done a task many times allows us to experience its variations and provides the ability to do them differently as a function of different environmental and internal conditions. In addition the Markov model provides an ability to simulate the various ways of doing the task by choosing its state values incrementally in an online manner.

The paper is organized as follows. Firstly, we elaborate the behavioral routine architecture. Then we describe two examples of a virtual agent solving natural tasks. The next section introduces the segmentation algorithm which partition a natural task into subtasks. Virtual agent's behavior is compared with that of humans in the following section. The next section presents a Markov graph which models the various execution orders in task solving by human subjects. Finally, we conclude and discuss future work.

The System

The behavioral routine architecture consists of a virtual reality platform which hosts a graphical human model and a virtual world, a set of software components controlling various actions of the virtual human, a human-world interface

which abstracts the agent's physical and visual interaction with the world, and a high speed image processing board which provides computational power required by real time vision.

A central problem in developing a vision enabled system is to produce realistic binocular images. To accomplish this task, we extended Boston Dynamics DI-GuyTM, a virtual reality human simulator package built atop SGI OpenGL PerformerTM, to produce a graphical human model with two 'eyes' in the form of virtual cameras attached to its head. This program can continuously generate images seen by the virtual human.

At startup, the system contains all the necessary information the virtual human needs to know about the world, such as what object is in the world and what characteristics they have. It uses a simplified frame-based knowledge representation. Then the systems loads a routine, interpret lines of commands and execute basic operations serially. A behavioral routine takes the form of a script, or a program, which has a simple grammar and uses a small set of commands such as 'remember' and 'saccade', and can also call other (sub)routines. The behavior of the virtual human is completely determined by this script, but different world settings may cause different results of the same routine.

A preprogrammed behavioral routine is loaded into the program and is parsed, interpreted, and executed line by line. The result of such execution is that commands are sent to direct the virtual human's eye movements and body movements. Such movements may cause the change of the view, so for every command sent out, a pair of new images, along with some deictic information such as location of fixation point, are received by the routine processor as updated input, closing the loop. Note that since this system is designed to use eye movement and embodiment, all visual input processing results in eye movements. For example, there isn't such command as "measure the distance between object A and object B", which requires 3D environment reconstruction. Such task is performed by first fixating on object A, memorizing its location, fixating on object B, memorizing its location and doing the calculation.

The current instruction set has not been completely designed, but the core members of it have been identified and implemented. Table 1 summarizes their functions. The first operation deserves special explanation. It's function is to search an object in binocular views, and computes the locations of this object relative to the virtual human. Since this research does not deal with low level vision, object recognition is considered as a black box. Only the input-output mapping matters, and its internal structure is beyond the consideration of this research. But for practical reasons, there must be some implementation of this operation to link the visual cognition with visual stimuli, so we added constraints to the scene and made object recognition easy: every object is uniquely color coded. In a noise free virtual environment, color matching suffices for object recognition. With more sophisticated implementation, less trivial recognition can be done, however the high level behavior will remains the same to a large extent.

The first three operations can be combined to do fixation.

Operation	Function	Parameter(s)
SearchObject	Find an object in both views	Object characteristics
Saccade	Conjugate eye movement	Yaw, pitch and roll
Vergence	Disconjugate eye movement	Angle
ImageMatching	Calculate two images' similarity	Labels of remembered images
Memorize	Remember a position or an image	Label to be assigned
MoveHand	Hand movement	Current fixation point
TurnHand	Turn hand (and object in hand)	Yaw, pitch and roll
Pickup/Dropoff	Object pickup/dropoff by hand	None

Table 1: Visual and motor operations implemented

SearchObject gives the object's 2D coordinate in both views. These data is used to initiate saccadic eye movements followed by vergence. This series of operations brings an object to the fovea.

An empirical study by Hancock et al. (Hancock, Baddeley, & Smith 1992) shows that principal components of natural images resemble Gaussian derivatives. Freeman and Anderson (Freeman & Adelson 1991) systematically introduced what they called *steerable filters*, a set of basis filters which could be linearly combined to synthesize filters of arbitrary orientations. Our image matching operation is performed on images filtered with 1st and 2nd order filters.

Using steerable filters introduces heavy computational load: huge amount of convolutions, which makes such a real time vision seem impractical for pure software systems. With special hardware, this hurdle can be cleared easily. Datacube MaxRevolution™, a pipelined image processing board, has a Xilinx Virtex II™FPGA chip with 128 MB memory. A typical batch convolution¹ takes as short as 6 milliseconds on this board.

Routine Controlled Virtual Human

In this section, we describe two experiments with natural tasks. These experiments use a virtual office/kitchen setting, where simple tasks are accomplished by the virtual human with vision and body movements. To demonstrate the system's validity, we compare its output with actual measurements performing the same tasks.

Coffee Pouring

This task involves picking up a coffee carafe, holding the carafe above the mug, tilting the carafe, pouring the coffee into the mug, monitoring the level of coffee in the mug, tilting the carafe back when the cup is almost full, and putting the carafe away.

The only visual behavior in this experiment is staring at the rim of the mug and send a signal to stop pouring when the coffee is full. This is done by calling two basic operations: memorization, and image matching. There is a training session at the beginning of the program, which makes up a scene with a full mug. This mug is remembered in the visual memory and is compared against right after the pouring

¹In our test we used 5 10×10 kernels on a 200×180 three channel color image.



(a)

Figure 1: Coffee Pouring: key moment. The mug is getting filled. The virtual human fixates at the rim of the mug, while his right hand is pouring coffee into the mug. Coffee in the carafe is not modeled for the sake of simplicity. The level of the coffee in the mug gradually rises and when it is about to full, the difference between the current view and a remembered image of a full cup drops below a certain threshold. Once that occurs, a motor command is sent to turn the right hand and pouring stops.

begins. Once the matching is successful, a motor command is sent to the human to turn its right hand. Since the carafe is grasped by hand, it's tilted back to original orientation and pouring stops. As introduced before, the image matching is performed in a filtered space on the Datacube.

Experiments with real human subjects suggests that human perform this task in the same way. Specifically, there are two important observations. First, almost all ² human subjects keep fixating at the mug rim until it is full and the next action is initiated. This indicates that vision is fully allocated during pouring. The other observation is that, each subject has a high consistency in the criterion for the *fullness* of a mug, suggesting that everybody has a single mental image of a *full mug* (Fig 2). The standard deviation is very small also suggesting that they are using a fixed algorithm. The behavioral routine exhibits the same structure, but

²We did the experiment with 15 subjects, and only one made a quick saccade off the rim during pouring.

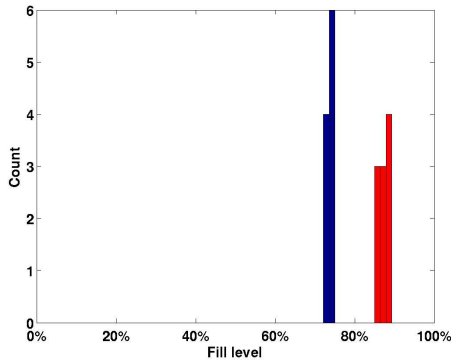


Figure 2: How full is full? We did experiments of pouring coffee into a cup on 3 subjects with 10 repetitions. We measured how full the coffee was after each pouring finished by calculating the ratio of the height of coffee level to the height of the cup rim. The mean values of the fill level for the subjects are 0.9037, 0.6690, and 0.8728 respectively, and the standard deviations are 0.0166, 0.0133, 0.0103 respectively. The red (right) histogram shows the data of subject 3, while the blue (left) one shows that of the virtual human, which has a mean of 0.7366 and a standard deviation of 0.0059.

the variance of the same measurement of the virtual human pouring coffee is about half that of human, suggesting the human brain may have to deal with higher noise levels.

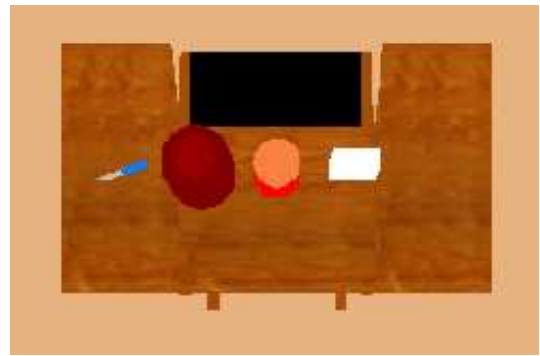
The physical interaction between the virtual human and the world, is made possible by motor operations such as *pickup* and *dropoff*. He has no knowledge about where objects are. When he receives a pickup command, for instance, he detects whether there is an object nearby. If yes, pick it up. In case of multiple candidates the closest one is chosen. Both scene traversal and object manipulation are part of the functionalities of the underlying virtual reality platform.

Peanut Butter and Jelly Sandwich

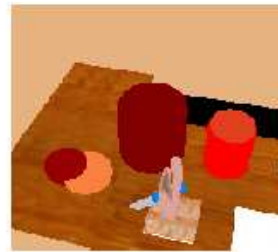
In a second experiment, a virtual peanut butter and jelly sandwich is made by the system and compared to real sandwiches made by humans. As shown in Fig. 3(a), this task involves identifying multiple objects, guiding hand movement using binocular vision and manipulating objects using hands. Firstly all the ingredients have to be recognized, and all the lids need to be removed. Then the virtual human spreads peanut butter on one slice of bread and jelly on the other. Finally he joins the two slices.

Basic visual operations used in this task include saccade, vergence, and object identification. As described earlier, a subroutine *locate* can be constructed by combining these operations. A segment of the behavioral routine for this task is as follows, in which *locate* refers to this subroutine. This segment finds and fixates on a loaf of bread, allocates a memory slot to store the location of the bread, and moves right hand to the fixation point.

```
locate bread
remember loc_bread
```



(a)



(b)



(c)

Figure 3: Sandwich making. (a) shows the experiment setting. From left to right: a knife with blue handle, jelly jar, peanut butter bottle, a loaf of bread, which consists of five separate slices. (b) and (c) are the agent's binocular views when spreading peanut butter on bread. The agent has already removed the jelly jar's and the peanut butter bottle's lids (on the desk), and is using a knife to spread peanut butter on a slice of bread. The hand doesn't seem to be holding the knife because, (1) in DI-GUY™ hands don't have movable fingers, and the last joint along the arm is the wrist, and (2) The virtual human has no knowledge about the shape of objects, thus does not know how to turn his hand to grasp it, and where to grasp.

puthand right

Embodied cognition plays an extremely important role in this task. The virtual human picks up objects in the world by making a fixation to an object and move the hand to the fixation point. He has minimal knowledge about the world. All he knows is how objects are color coded, e.g. bread is white, knife handle is blue, etc.. These information helps skipping low level vision such as view segmentation and object recognition. The virtual human knows little about the geometry of the scene, however, he can interact with the world by utilizing the information embodied in his body, rather than entirely relying on vision and the reconstruction of an internal 3D model of the world.

With embodiment, the virtual human can know accu-

rately where the fixation point is by computing trigonometry with interocular distance and eye orientations, using $e/2 = d \tan(\alpha/2)$ where e is the interocular distance, d the distance between the fixated object and the middle point between the two eyes, and α the vergence angle. Similarly, locating an object which is held by hand is automatic: no visual search is need, the current body posture³ tells where it is, and one saccade is enough to bring that object into the fovea. The fixation pattern of the virtual human in the sandwich making task is illustrated in Figure 5(a).

Automatic Segmentation of Sequential Behavior

Compared with deterministic computer agents, humans solve everyday tasks with high variety. For instance, different people make sandwiches in different ways. It is important to identify the similarity and difference between those solutions, in form of behavioral routines. We started with identifying local patterns in behavioral routines, and developed an algorithm to find common subsequences in different routines. The purpose is to partition sequential behavior into segments such that each behavior can be represented as an assembly of these segments in a certain order. Each behavior is a sequence of behavioral primitives and solves a same task. Since the actually planning in such a solution varies across individual subjects, this algorithm attempts to identify common structures and patterns in different solutions, and to model natural behavior in a two level hierarchy, cognitive macros and cognitive primitives.

Since this algorithm assumes a routine based modeling of cognitive behavior, in which complex behavior is programmed with a fixed set of primitives. So the problem reduces to finding a small common segmentation of a set of strings which share a finite alphabet. By small we mean the number of segments is close to minimum. For example, given input strings *goodmorning*, *morningcall* and *goodcall*, we want to automatically identify segments *good*, *morning* and *call*. To simply this problem, we assume the lengths of these strings are equal, and all the segments appear in every input string.

The input to the algorithm is a series of strings $\{B^{(i)} | i = 1, 2, \dots, N, B^{(i)} \in \Sigma^L, L > 0\}$, where Σ is the alphabet. The output of the algorithm is $\{S^{(j)} | \forall B^{(i)}, \exists p \in P(M) : B^{(i)} = S^{(p_1)} + S^{(p_2)} + \dots + S^{(p_M)}\}$, where $P(M)$ is the permutation of integers $[1, M]$ and $+$ is the string concatenation operation.

Our algorithm has two steps. Step 1. Find all the potential segments. We pick an arbitrary input string $B^{(i)}$, compare it with another input string $B^{(j)}$, $i \neq j$, and collect all the segments in $B^{(i)}$ that also exists in $B^{(j)}$, by shifting and matching. Here the location of such segments in $B^{(i)}$ is important: identical substrings in different locations are considered different segments. Repeat this process on all other input strings $B^{(j)}$, $i \neq j$, and compute the intersection

³We don't maintain all the joint angles and physical constraints of the body at this time, but calculating the location of hand is replaced by a query in the virtual reality program.

of collected segments. Now we get all the segments in $B^{(i)}$ that also exists in *all* other strings, i.e. the solution to the automatic segmentation problem consists segments selected from this collection. We denote this candidate collection as $C = \{C_i | i = 1, 2, \dots, K\}$. This step has a time complexity of $O(L^2N)$.

Step 2. Search a combination of candidate segments in C . This is an NP Complete problem. Our approach is a knapsack algorithm which doesn't guarantee an optimal solution, but generates a good one without having to exhaust all the combinations. The basic idea of this greedy algorithm is to give priority to long candidate segments when looking for a combination. First we sort C by length decreasingly, and get C^* . Then the algorithm is a depth first search of a binary decision tree in which the root nodes correspond to the decision "Is the longest candidate segment C_1^* in the segmentation?", and all i th layer nodes correspond to the decision "Is i th longest candidate segment C_i^* in the segmentation?". As the depth first search traverse proceeds, the combination of selected candidates, decided by the path from the root, is tested for the qualification of a solution to the problem, i.e. whether the current combination make a segmentation of *each* input string $B^{(i)}$. Once this criterion is met, the search stops.

This algorithm doesn't guarantee an optimal solution and each run and generate different segmentation. Since the complexity of optimization is exponential to the length of input strings, we use the greedy algorithm when dealing with long inputs and go back to brute force when input strings are short.

We applied a few tricks to accelerate the algorithm. For example, we compressed the input routines by encoding each primitive name with one letter, and skipped all the separators. We tested this algorithm with visuo-motor routines describing a virtual human making sandwiches and the algorithm successfully identified ten segments, i.e. ten subtasks which can be solved with different orders. This segmentation, given in Table 2, is the basis of the next two sections.

Human Data

We recorded a few human subjects making peanut butter sandwiches. The experimental setting is shown in Fig 4: objects are placed in the same positions as in the virtual reality. We used an SR Research EyeLink®II eye tracker to record gaze movement. We analyzed the transitions of fixation points and the fixation pattern is illustrated in Fig 5(b).

Human eye movement is far more noisy and unpredictable than camera movement in a computer vision system. We processed human data by clustering fixation points and associating them to objects in the scene. Fortunately this association is easy: Almost every fixation point is close to a certain object which is relevant to the task. From the similarity between the two fixation pattern in Fig 5 we can see that the design of instruction set in our behavioral routine architecture provides functional support for generating human like visual behavior. Furthermore, it has the extensibility and flexibility to model human world interaction in natural tasks, and provides a platform to study complex human



Figure 4: Experiment setting. A subject is making a peanut butter sandwich with real objects which have identical positions as in the virtual reality.

visuo-motor behavior.

Table 2 summarizes the scheduling of 10 subtasks in making a peanut butter sandwich by 3 human subjects⁴. Despite that some chronological constraints, e.g. BT, PLF and KH must precede POB and JOB, have ruled out most of the 10! orders, the number of possible orders remaining is still a large number⁵. However, experiments with much more subjects show that the orders picked by human subjects display some common features. For instance, BT is always the starting point and KH is always immediately followed by POB or JOB. The psychophysical and social reasons underlying these special constraints and the mechanism of scheduling these subtasks can be studied in the routine based framework of visuomotor interaction between the agent and its surroundings.

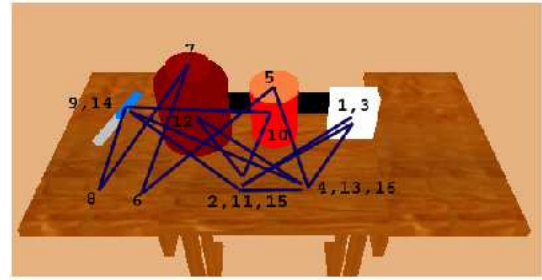
Markov Model for Task Planning

One obvious observation is, different subjects have different orders in solving each subtask, but there are some statistically common patterns, e.g. KH is almost always followed by one of the two spreading subtasks, POB or JOB. This motivates us to use a Markov model to capture, and later generate, these patterns.

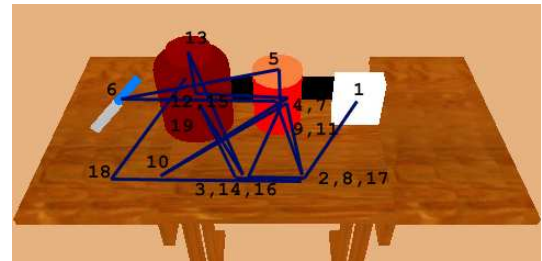
Generally, we denote these subtasks as $T = \{t_i | i = 1, 2, \dots, N\}$. First we compute the power set of the $P(T) = \{\emptyset, \{t_1\}, \{t_2\}, \dots, \{t_1, t_2\}, \dots, T\}$, and then construct a graph in which the vertex set $V = P(T)$, and edge set $E = \{\langle p_i, p_j \rangle | p_i, p_j \in V, p_i \subset p_j, |p_j - p_i| = 1\}$. In plain words, each vertex of this graph is a set of subtask that have been done, indicating the progress in solving a com-

⁴We make some assumptions such as the knife is picked up only once and is not put down until spreading finishes.

⁵If we divide the 10 subtasks into 3 stages: $\{BT, PLF, JLF, KH\}$, $\{POB, JOB\}$ and $\{PLO, JLO, KT, FB\}$, we have $4! \times 2! \times 4! = 1152$ different orders



(a)



(b)

Figure 5: Fixation Patterns in Sandwich Making. The numbers indicate order of fixations. Some points have multiple labels because of revisits. (a) is the fixation pattern of the virtual agent, which is controlled by handmade script, while (b) shows that of a human subject making sandwich with the same setting. Each fixation point is one of the following 10 positions. Upper row: knife, jelly bottle, jelly lid, peanut butter bottle, peanut butter lid. Lower row: the places to put jelly lid, peanut butter lid, left slice of bread, right slice of bread.

plete task, while each edge is the transition between two statuses, one of which have exactly one more subtask done. This graph is a directed acyclic graph (DAG). The vertex \emptyset is the source of this graph, and vertex T is the sink. Any path from source to sink passes exactly $N+1$ vertices, and N edges.

Note that in the construction of such a DAG, constraints about chronological order of subtasks apply, as shown in Fig 6. For example, POB always happens after PLF, and before FB. With these constraints, the number of vertices is narrowed down from 2^{10} to 41.

Then, we feed data into this Markov model, and calculate transition probabilities between statuses. With these probabilities, subtask orders and be generated probabilistically. An example is as follows.

With input data:

	1	2	3	4	5	6	7	8	9	10
BT	abc									
PLF		a	c		b					
JLF		bc				a				
KH			ab	c						
POB				a	c	b				
JOB				b		c	a			
PLO					a				b	c
JLO									c	ab
KT							c	ab		
FB							b	c	a	

Table 2: Scheduling of subtasks. The task is decomposed into ten subtasks including BT (putting bread on table), PLF (taking peanut butter lid off), JLF (taking jelly lid off), KH (grabbing knife in hand), POB (spreading peanut butter on bread), JOB (spreading jelly on bread), PLO (putting peanut butter lid on), JLO (putting jelly lid on), KT (putting knife on table), and FB (flipping bread to make a sandwich). Letters a, b and c denote the orders of subtasks taken by 3 subjects, e.g. in the first 2 steps subject c put bread on the table and took jelly lid off.

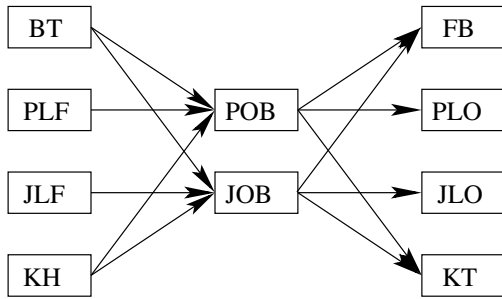


Figure 6: Dependencies among subtasks. Arrows connecting subtask t_i and t_j denotes that t_i must be done prior to t_j .

Subject 1	0	3	1	4	7	2	5	6	9	8
Subject 2	2	0	1	3	4	5	9	6	8	7
Subject 3	0	2	3	5	1	4	6	9	7	8
Subject 4	0	2	1	3	4	5	9	6	8	7

the most probable path is:

(75%) 0 \rightarrow (67%) 2 \rightarrow (67%) 1 \rightarrow (100%) 3 \rightarrow (100%) 4 \rightarrow (100%) 5 \rightarrow (67%) 9 \rightarrow (100%) 6 \rightarrow (67%) 8 \rightarrow (100%) 7, with probability 14.8

Conclusions and Future Work

Inspired by visual routines model, we developed a computer vision architecture which takes an active and situated approach, focuses on high level vision, and exploits embodied information. Unlike any previous work, it has an basic operation set with which virtually any visuo-motor behavior can be constructed, and has the potential to model complex human world interaction as programming with perceptual and motor primitives.

Basing on the sequential modeling of cognition and agent-world interaction, we introduced an algorithm to automatically identify top level structure of behavioral routines by finding common segmentations across different routines, and dividing a task into subtasks.

Our experiments show that humans make the same kinds of trade-offs that are made in the visual routines model. In coffee pouring, individual subjects stop at a fixed distance from the top of the cup and have very small deviations between the level in successive fills, suggesting the use of standard procedures. Also, as shown by our model, most of this variances can be explained as noise in an image matching process. In sandwich-making, individual subjects exhibit very similar task orderings so that their performance can be easily captured by a behavioral routines based Markov Decision Process that contains the alternate task orderings.

We also described a Markov model which can capture different orders used by human subjects to solve subtasks. This model can be used to generate behavioral routines that solve the same task in a human like fashion.

As future work, we are interested in online behavior recognition by parsing head-hand-eye movement data, along with videos taken by a head-mount camera, into behavioral routines. With the resulting routines, we will be able to recognize the status of task solving and approximate the agenda of subjects.

References

- Ballard, D. H. 1991. Animate vision. *Artificial Intelligence* 48:57–86.
- Beetz, M.; Arbuckle, T.; Cremers, A. B.; and Mann, M. 1998. Transparent, flexible, and resource-adaptive image processing for autonomous service robots. In *European Conference on Artificial Intelligence*, 632–636.
- Chapman, D. 1991. *Vision, Instruction, and Action*. MIT Press.
- Findlay, J. M., and Gilchrist, I. D. 2003. *Active Vision: The Psychology of Looking and Seeing*. Oxford University Press.
- Freeman, W. T., and Adelson, E. H. 1991. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13(9):891–906.
- Hancock, P. J.; Baddeley, R. J.; and Smith, L. S. 1992. The principal components of natural images. *Network: Computation in Neural Systems* 3(1):61–70.
- Horswill, I. 1995. Visual routines and visual search: a real-time implementation and an automata-theoretic analysis. In *Proc., Int'l. Joint Conf. on Artificial Intelligence*, 55–63.
- Johnson, M. P.; Maes, P.; and Darrell, T. 1994. Evolving visual routines. In *Artificial Life IV, Proc., 4th Int'l Workshop on the Synthesis and Simulation of Living Systems*, 198–209. MIT, Cambridge, MA, USA: MIT Press.
- Kahn, R. E., and Swain, M. J. 1995. Understanding people pointing: The perseus system. In *Proc., Int'l Symp. on Computer Vision*, 11A Motion III.

- Noton, D., and Stark, L. 1971. Scanpaths in eye movements during pattern perception. *Science* 171(3968):308–311.
- Salgian, G., and Ballard, D. H. 1998. Visual routines for autonomous driving. In *Proc., Int'l. Conf. on Computer Vision*, 876–882.
- Ullman, S. 1996. *High-level vision*. MIT Press.
- Wilson, M. 2002. Six views of embodied cognition. *Psychonomic Bulletin and Review* 9:625–636.
- Yarbus, A. L. 1967. *Eye movements and vision*. New York, Plenum Press.