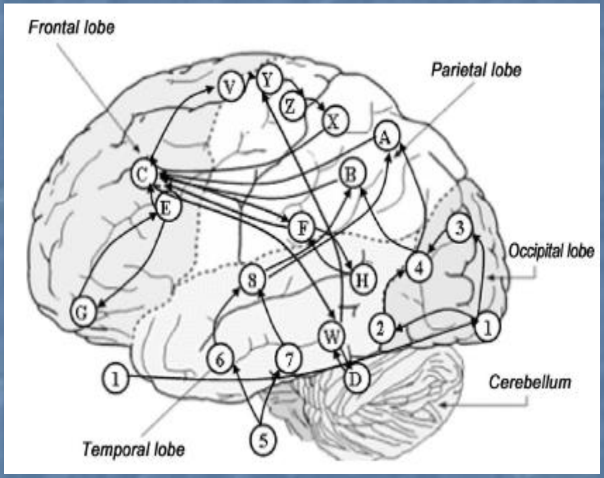
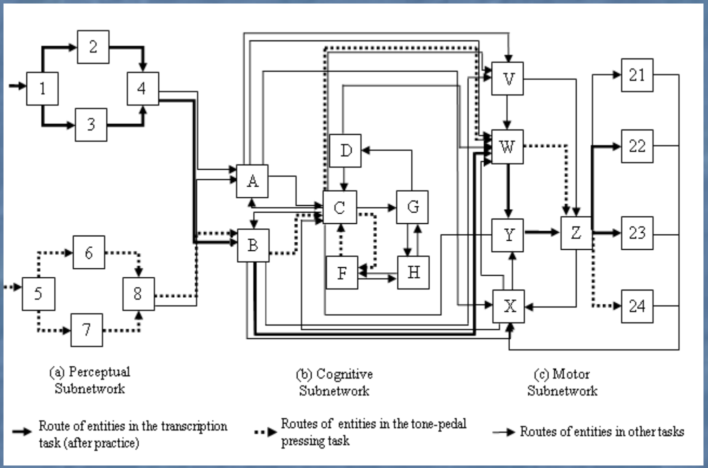
Integrative Modeling and Simulation of Human Behavior and Human-Machine Systems with Queuing Network (QN) Architecture





Queueing Network of Mental Architecture

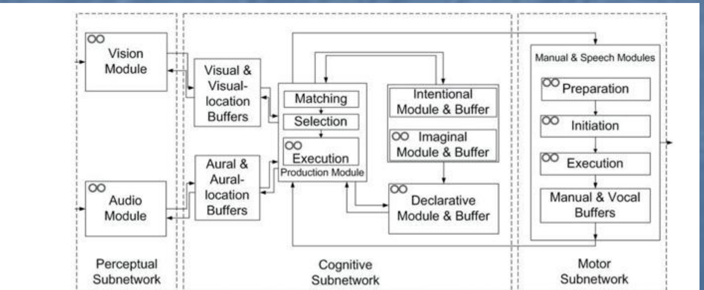


Figure 3. Server structure of QN-ACTR. Queue symbols (shown as two circles) mark the servers where queues are added from the QN's perspective. All the server processing logics in the QN-ACTR are identical to the corresponding algorithms in ACT-R (adapted from Cao & Liu, 2012c).

Yili Liu UM-IOE HFES-Aspire Workshop 2024

Thanks to All My

Collaborators,

Critics,

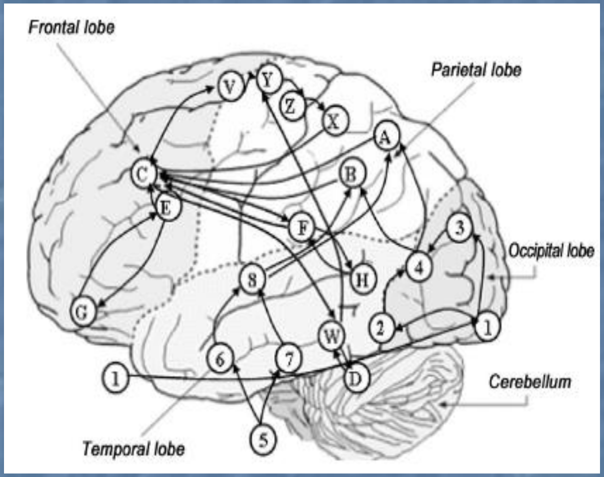
Observers,

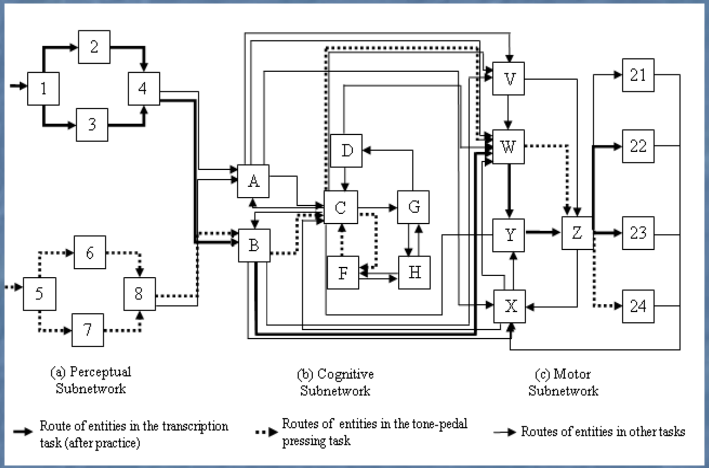
Reviewers,

Sponsors,

Supporters!

Integrative Modeling and Simulation of Human Behavior and Human-Machine Systems with Queuing Network (QN) Architecture





Queueing Network of Mental Architecture

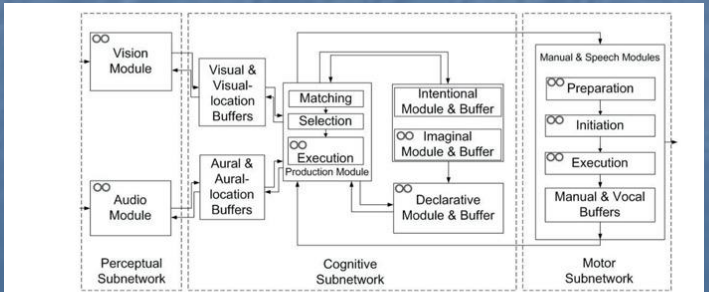


Figure 3. Server structure of QN-ACTR. Queue symbols (shown as two circles) mark the servers where queues are added from the QN's perspective. All the server processing logics in the QN-ACTR are identical to the corresponding algorithms in ACT-R (adapted from Cao & Liu, 2012c).

Queueing Network (QN)
Models of Human Behavior (MHB)
QN-MHB

1. RT: Reaction Time (QN-RT) and Mental Structure
2. RT and Accuracy (QN-RMD) (Mental Structure vs State of Mind)
3. Procedural Tasks (QN-MHP or QN-MHP-BE)
4. Complex Cognition Tasks (QN-ACTR)
5. Visual Attention tasks (QN-NSEEV) (QN-RLEM)
6. Manual or Continuous Control tasks (QN-Control: Classical/Modern)
7. Basic Body Motion tasks (QN-MTM)
8. Mind-Body System (QN-MBS)
9. Neural level (QN-Neural: Indexes and Neural Networks)
10. Nervous and Endocrine Systems (QN-NES)
11. Multi-Person Multi-Machine QN (QN-HMN)
12. Engineering Applications in various domains

Note: relations with **Task Network** Methods/Tools such as Micro Saint #, IMPRINT)

Queueing Networks (QN) are everywhere (as phenomena)

- Airports
- Hotels/restaurants/conference rooms
- Manufacturing facilities
- Traffic networks (road, air, rail,…)
- Computer networks
- Inside computer (chips, ,memory, CPUs)
- Shopping malls
- You name it…******
- Inside our mind and body and

 in their interactions with human-machine-environments

 --The theme of this Workshop

Queueing Networks (QN) are **powerful** (as a method)

- Intuitive
- Mathematical
- Computational
- Simulational
- "Generative"
- Versatile
- Widely used
- Integrative

Queueing Networks (QN) are easy-to-understand (the concepts)

- Servers
  (e.g., service time, capacity, busy time)
- Customers (Entities)
  (e.g., arrival time/rate, departure time, sojourn time, “balk rate”)
- Routes/Paths and Routing Probabilities

Queueing Network (QN)
Models of Human Behavior (MHB)
QN-MHB

1. RT: Reaction Time (QN-RT) and Mental Structure
2. RT and Accuracy (QN-RMD) (Mental Structure vs State of Mind)
3. Procedural Tasks (QN-MHP or QN-MHP-BE)
4. Complex Cognition Tasks (QN-ACTR)
5. Visual Attention tasks (QN-NSEEV) (QN-RLEM)
6. Manual or Continuous Control tasks (QN-Control: Classical/Modern)
7. Basic Body Motion tasks (QN-MTM)
8. Mind-Body System (QN-MBS)
9. Neural level (QN-Neural: Indexes and Neural Networks)
10. Nervous and Endocrine Systems (QN-NES)
11. Multi-Person Multi-Machine QN (QN-HMN)
12. Engineering Applications in various domains

    Note: relations with **Task Network** Methods/Tools such as Micro Saint #, IMPRINT)

Mathematical Models of RT and Mental Structure Classified in terms of Discrete versus Continuous Information Transmission and Serial versus Network Architecture

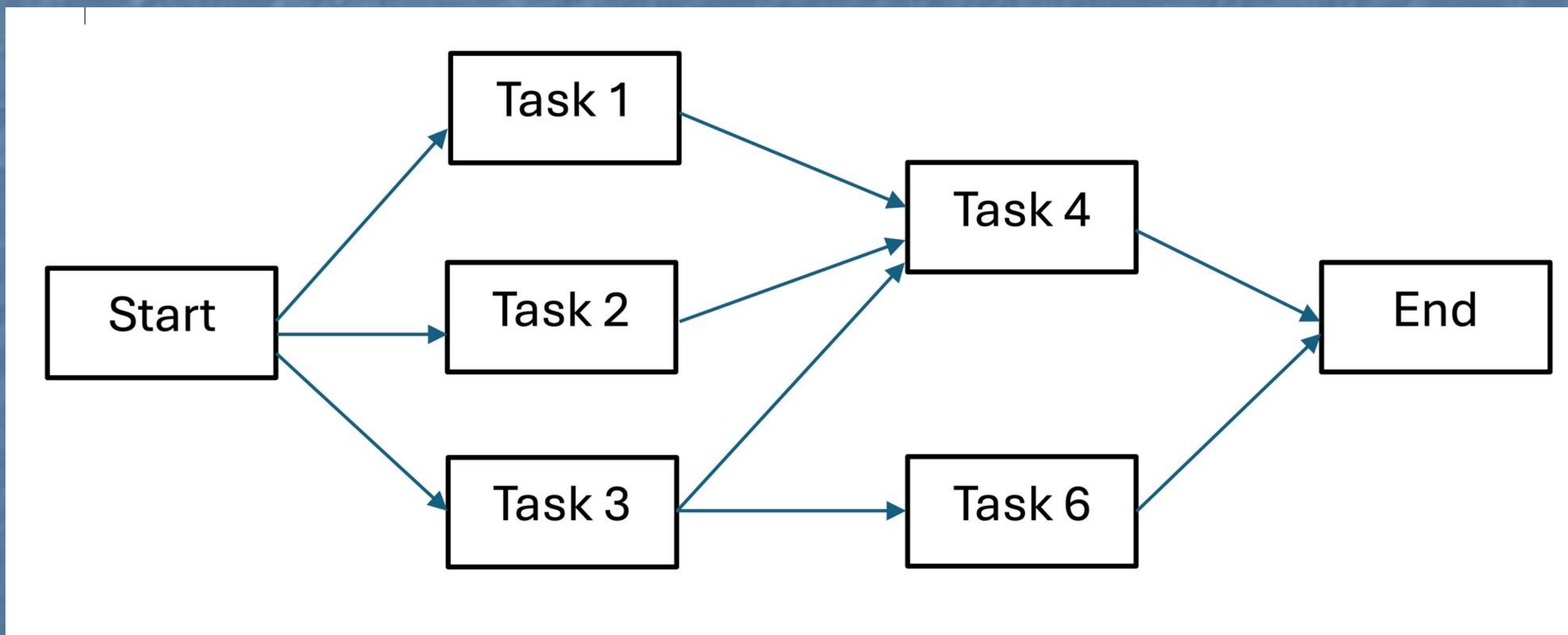
Mathematical Models of RT and Response Accuracy (sequential sampling models)

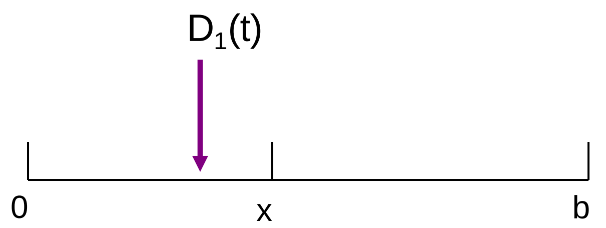
(from Liu, 1996, "Queueing network modeling of elementary mental processes," Psychological Review, 103(1), pp. 116-136).

	Architectural arrangement of mental processes		
Temporal Transmission	Serial Stages	Network Configurations	**State** transition
Discrete	Subtractive Additive factors General Gamma	Critical Path Network (PERT)	Counter/accumulator Random-walk
Continuous	Cascade Queueing series	**Queueing Network (QN)** ↔	Accumulator Diffusion **Reflected Multidimensional Diffusions (RMD)**

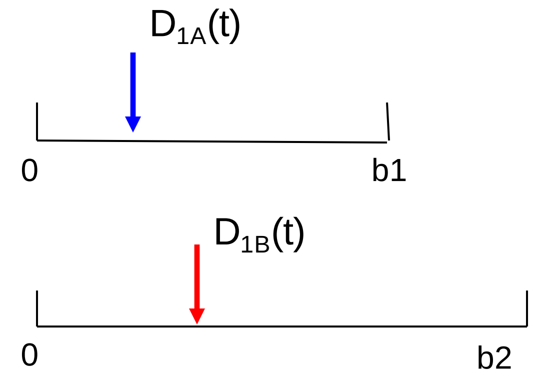
- **Non-unidirectional flow**
- **Non-selective influence**
- **By-passing**
- **Fixed capacity**

Task Network
(Project network, PERT, CPN, Mission Network, etc.)



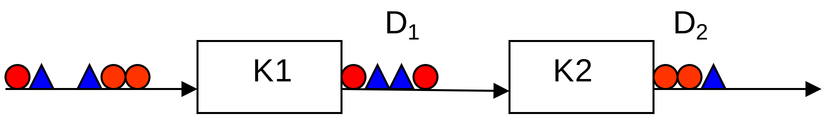


1. Random-walk/Brownian-Motion/Diffusion Model

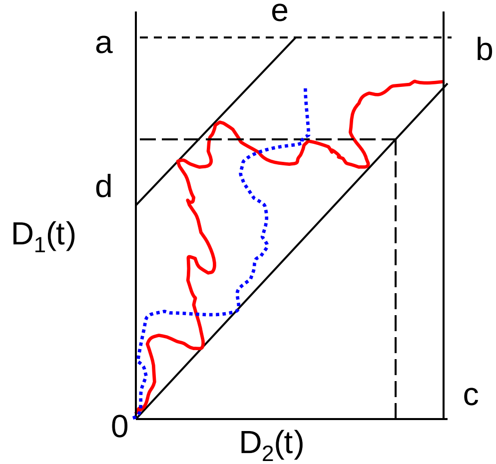


2. Counter/Accumulator Model

Diffusion and Accumulator Models of Speed-Accuracy Tradeoff (All 1-D)



a). A tandem two-server system with two types of customers: type A ("triangles") and type B ("circles")



b). 2-D diffusion representation of $\{D_1(t), D_2(t)\}$ in a continuous tandem queue

Queueing Network-Model Human Processor (QN-MHP): A Computational Architecture for Multitask Performance in Human-Machine Systems

YILI LIU, ROBERT FEYEN, and OMER TSIMHONI
University of Michigan

Queueing Network-Model Human Processor (QN-MHP) is a computational architecture that integrates two complementary approaches to cognitive modeling: the queueing network approach and the symbolic approach (exemplified by the MHP/GOMS family of models, ACT-R, EPIC, and SOAR). Queueing networks are particularly suited for modeling parallel activities and complex structures. Symbolic models have particular strength in generating a person's actions in specific task situations. By integrating the two approaches, QN-MHP offers an architecture for mathematical modeling and real-time generation of concurrent activities in a truly concurrent manner. QN-MHP expands the three discrete serial stages of MHP, of perceptual, cognitive, and motor processing, into three continuous-transmission subnetworks of servers, each performing distinct psychological functions specified with a GOMS-style language. Multitask performance emerges as the behavior of multiple streams of information flowing through a network, with no need to devise complex, task-specific procedures to either interleave production rules into a serial program (ACT-R), or for an executive process to interactively control task processes (EPIC). Using QN-MHP, a driver performance model was created and interfaced with a driving simulator to perform a vehicle steering, and a map reading task concurrently and in real time. The performance data of the model are similar to human subjects performing the same tasks.

Categories and Subject Descriptors: H.1.2 [**Models and Principles**]: User/Machine Systems—*Human information processing, human factors*; I.6.5 [**Simulation and Modeling**]: Model Development—*Modeling methodologies*

General Terms: Human Factors

Additional Key Words and Phrases: Cognitive model, human-computer interaction, cognition, user interfaces, human information processing

R. Feyen is currently at School of Industrial Engineering, Purdue University.
Authors' address: Y. Liu, O. Tsimhoni, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109; email: {yililiu,omert}@umich.edu; R. Feyen, School of Industrial Engineering, Purdue University, West Lafayette, IN 47907; email: rfeyen@purdue.edu.

ACM Transactions on Computer-Human Interaction, Vol. 13, No. 1, March 2006, Pages 37–70.

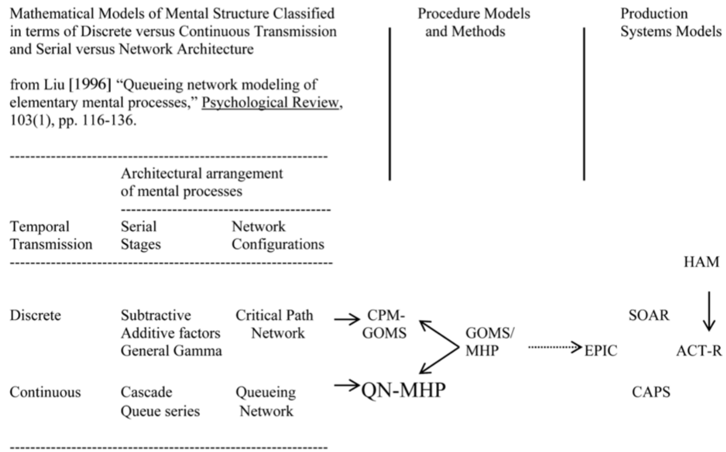
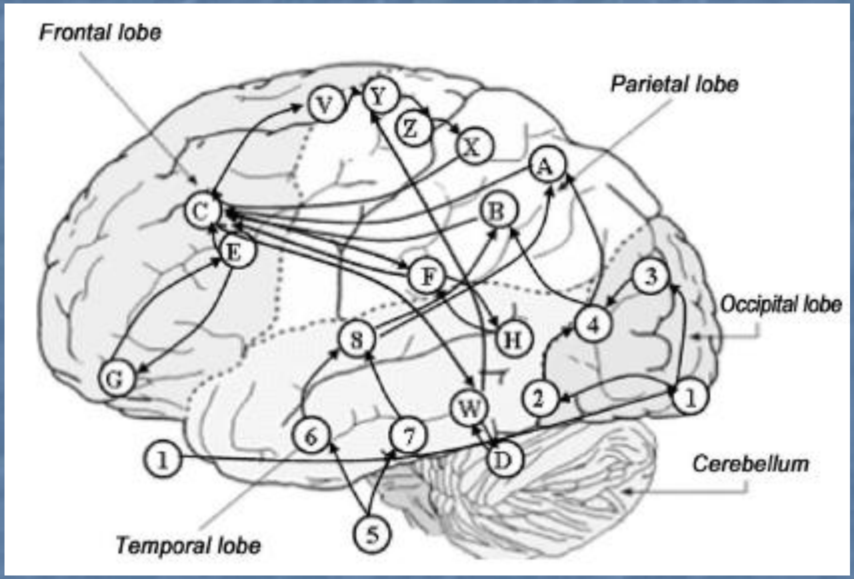
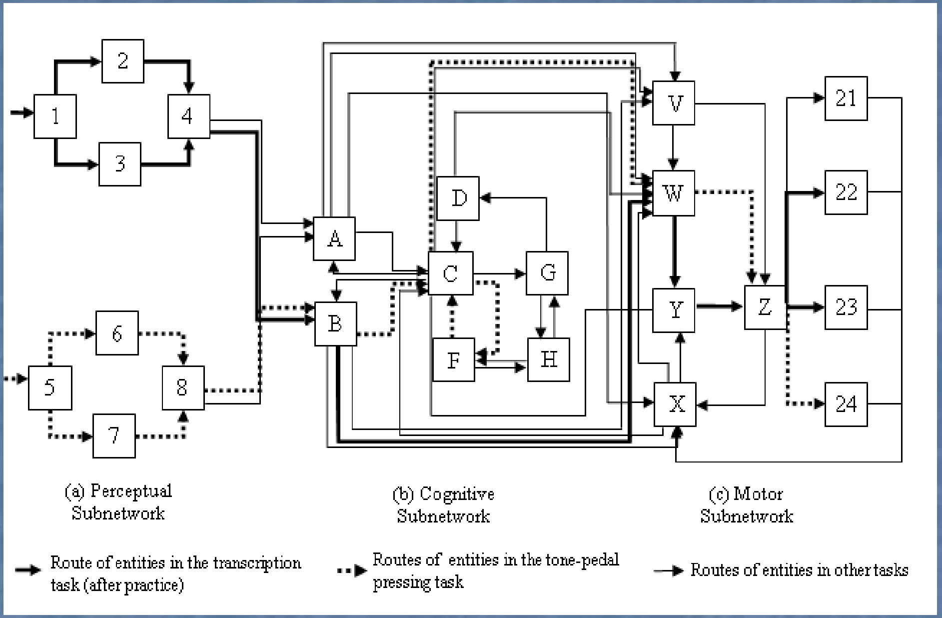


Fig. 1. Mathematical models of mental structure and procedure/production system models of cognitive architecture.

existing approaches. More specifically, we describe our current and proposed work in developing a complementary modeling approach that integrates the modeling philosophy and methods of the procedure-knowledge/production-systems models listed above and the mathematical/simulation theories and methods of queueing networks. Research on queueing networks is not only a major branch of mathematics and operations research but also one of the most commonly used methods for performance analysis of a large variety of real-world systems such as computer, communications, manufacturing, and transportation networks (e.g., Disney and Konig [1985]; Denning and Buzen [1978]; Boxma and Daduna [1990]). A large knowledge base on queueing networks exists, and some well-developed simulation and analysis software programs are widely used by engineers world-wide. Furthermore, from the psychological modeling perspective, as published in a *Psychological Review* article entitled "Queueing network modeling of elementary mental processes" [Liu 1996], we have successfully used queueing networks to integrate a large number of influential mathematical models of mental structure and psychological processes, such as Sternberg's serial stages model [Sternberg 1969], McClelland's cascade model [McClelland 1979], and Schweickert's critical path network model [Schweickert 1978] (see the left-half of Figure 1). From the systems engineering perspective, we have successfully used queueing networks to integrate single-channel one-server queueing models (e.g., Senders [1964]; Rouse [1980]) and the parallel processing models (e.g., Laughery [1989]; Wickens and Liu [1988]) as special cases [Liu 1994, 1997].





Human Brain

Queueing Network of Mental Architecture

QN-MHP-BE or QN-MHP

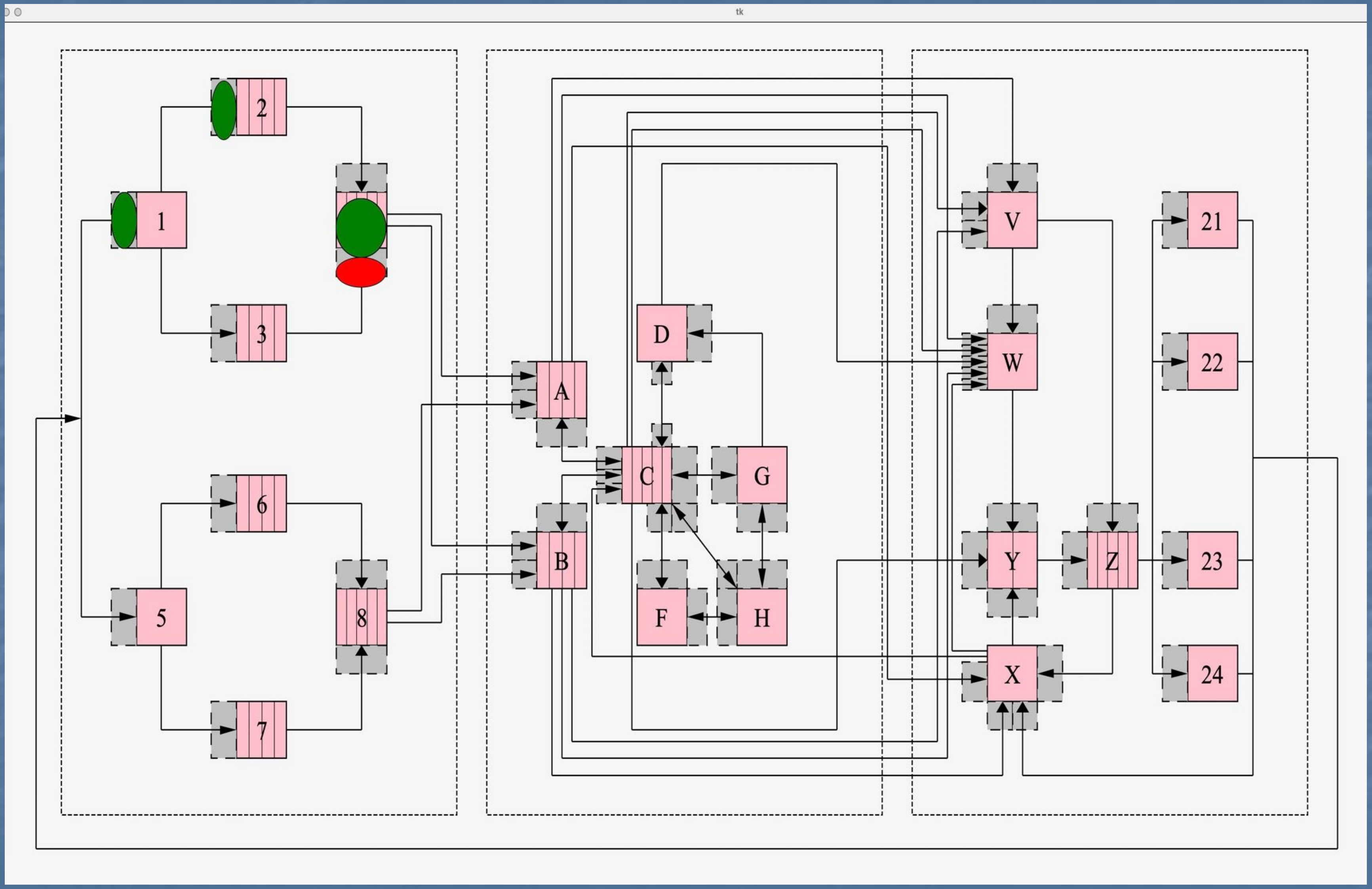
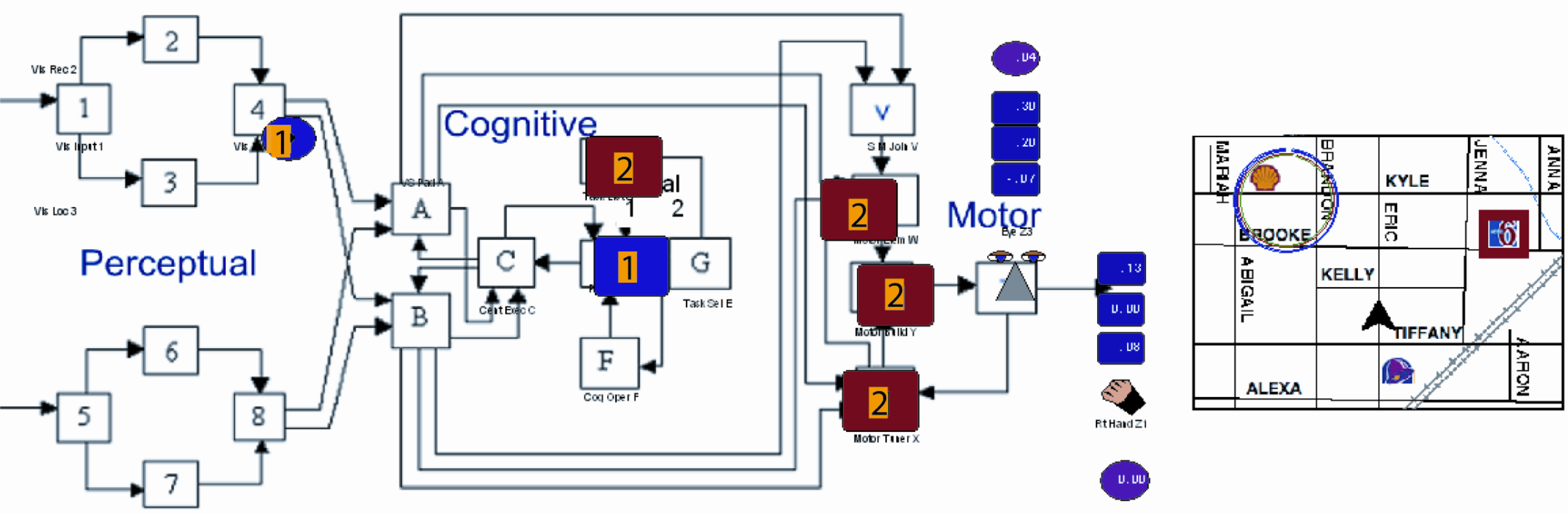
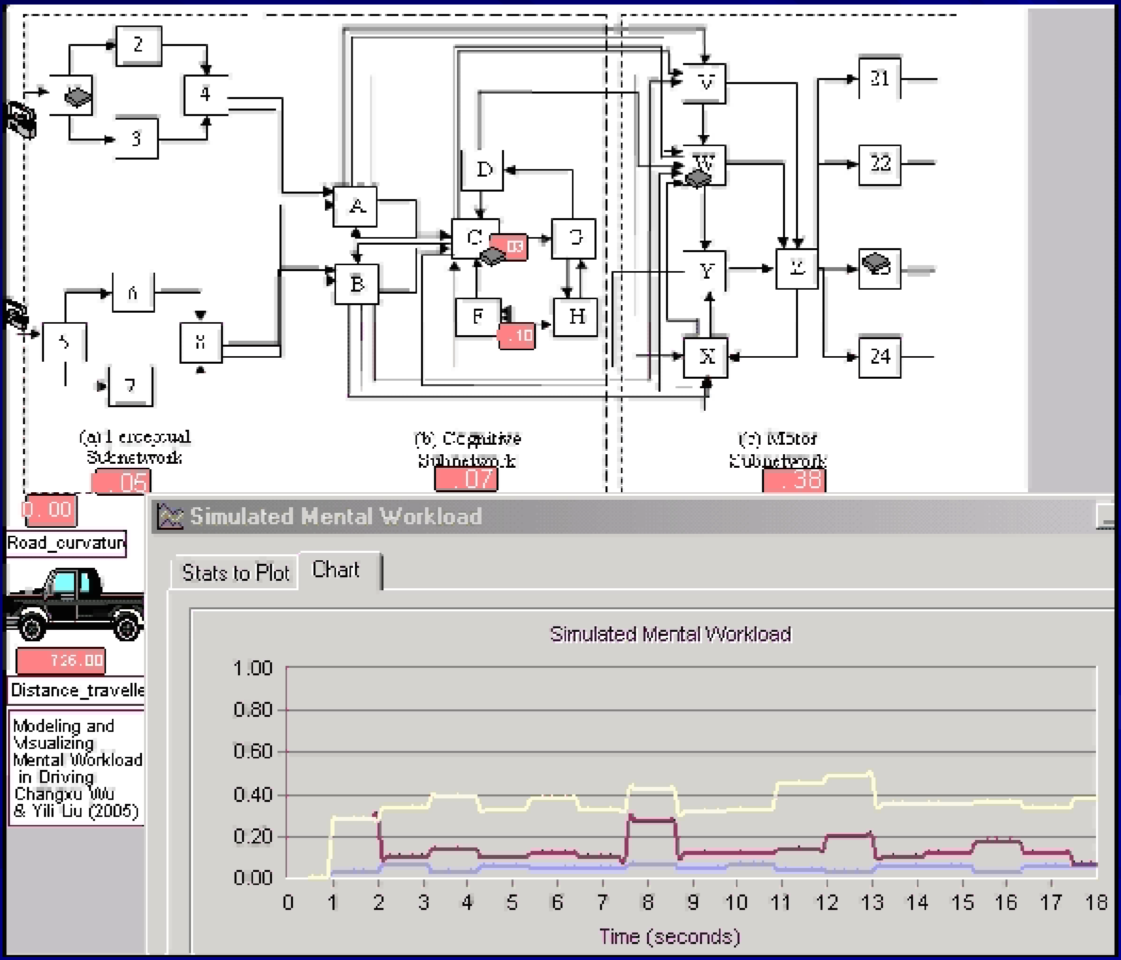


Fig. 10. Screenshot of QN-MHP in action. A visual entity is about to be processed by server A as two concurrent tasks are being processed in the cognitive subnetwork. The eye is looking at the map but a steering action is still underway.



Visualizing Mental Workload (perceptual, cognitive, motor loads)



QN-MHP

Software Demo

and

Code Analysis

INTL. JOURNAL OF HUMAN–COMPUTER INTERACTION, *25*(6), 554–581, 2009

ISSN: 1044-7318 print / 1532-7590 online
DOI: 10.1080/10447310902973182

Taylor & Francis
Taylor & Francis Group

QN-ACES: Integrating Queueing Network and ACT-R, CAPS, EPIC, and Soar Architectures for Multitask Cognitive Modeling

Yili Liu
The University of Michigan

Comprehensive and computational models of human performance have both scientific and practical importance to human–machine system design and human-centered computing. This article describes QN-ACES, a cognitive architecture that aims to integrate two complementary classes of cognitive architectures: Queueing network (QN) mathematical architecture and ACT–R, CAPS, EPIC, and Soar (ACES) symbolic architectures. QN-ACES represents the fourth major step along the QN architecture development for theoretical and methodological unification in cognitive and human–computer interaction modeling. The first three steps—QN architecture for response time, QN-RMD (Reflected Multidimensional Diffusions) for response time, response accuracy, and mental architecture, and QN-MHP (Model Human Processor) for mathematical analysis and real time simulation of procedural tasks—are summarized first, followed by a discussion of the rationale, importance and specific research issues of QN-ACES.

1. INTRODUCTION

The increasing complexity of advanced human–machine systems makes it necessary for system designers to consider human capabilities and limitations as early as possible in system design. In order to reduce risks associated with poor task design with appropriate tools and methods for task analysis and function allocation, it is important to develop models of human performance and human-system interaction that are comprehensive, computational, science-driven, and application-relevant.

Models of human performance and human–system interaction should be comprehensive to capture the whole range of concurrent perceptual, cognitive, motor, and communication activities of human-system performance. These models should be computational and computerized to allow quantitative and rigorous simulation and analysis of design alternatives and scenarios. These models should be science driven, with deep roots in and strong connections with cognitive science

Correspondence should be addressed to Yili Liu, Department of Industrial & Operations Engineering, The University of Michigan, 1205 Beal Avenue, Ann Arbor, MI 48109-2117. Email: yililiu@umich.edu

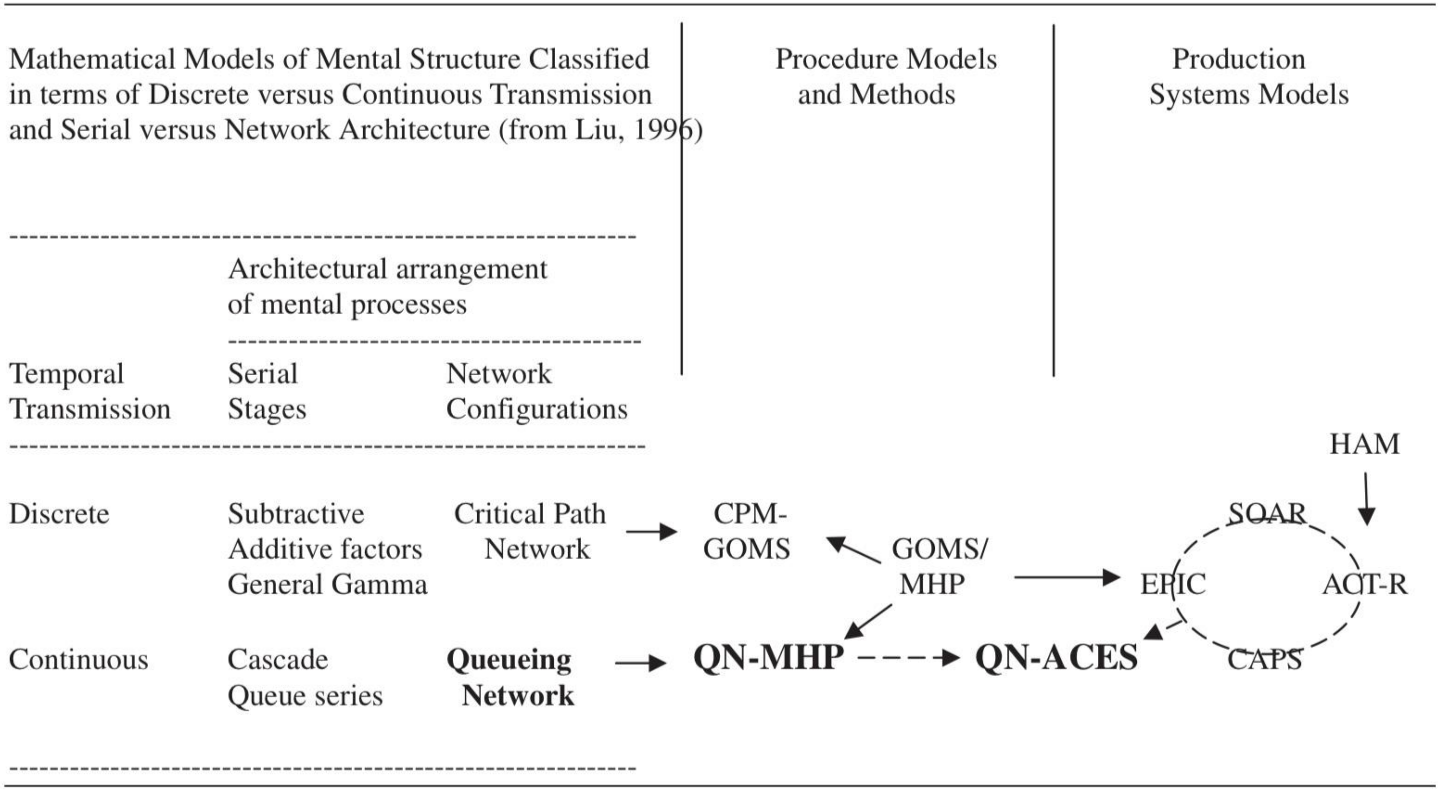
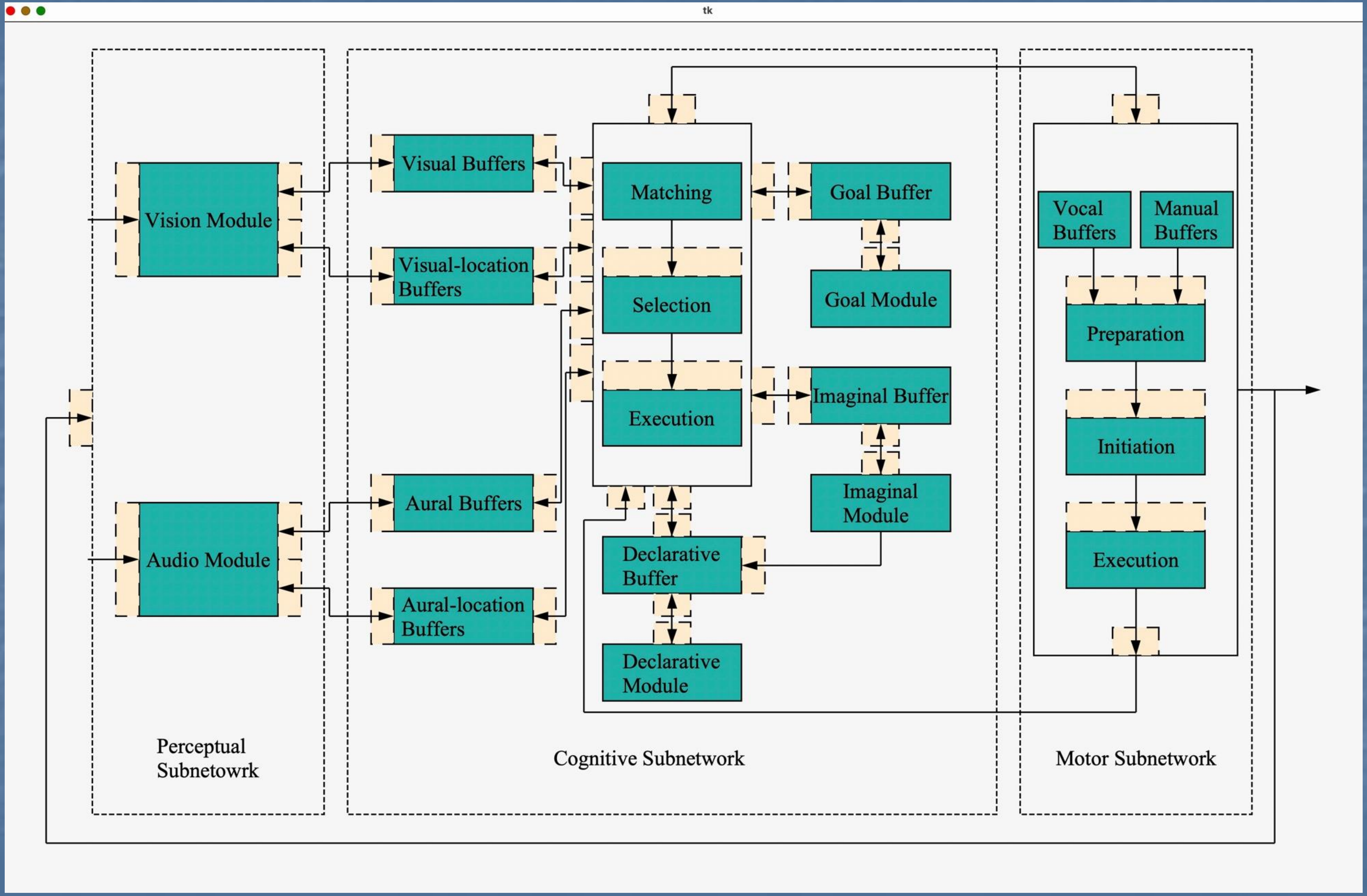


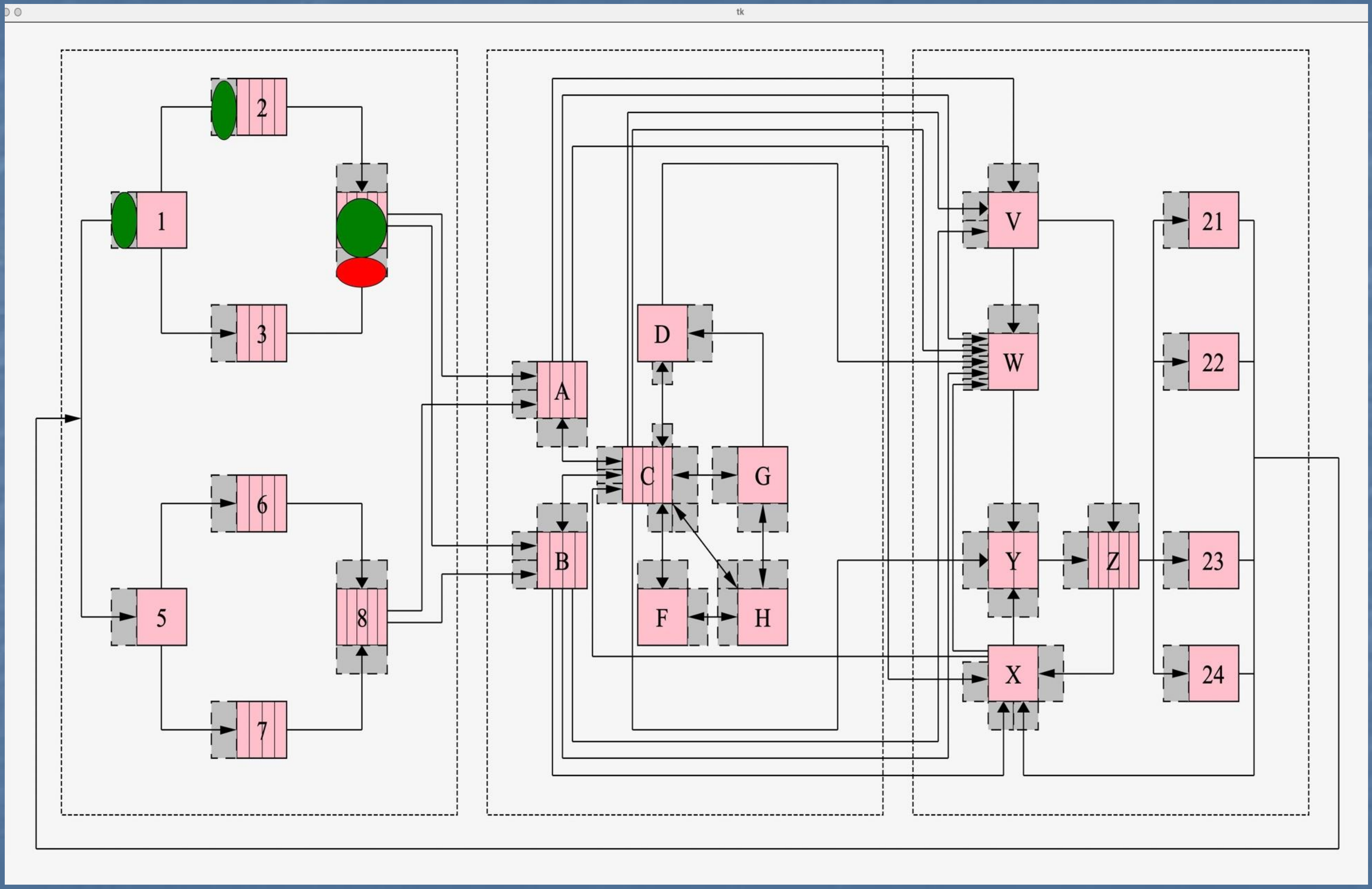
FIGURE 2 Mathematical and symbolic models of mental architecture (Liu, 2006; Liu et al, 2006) showing the relationship between QN, QN-MHP, QN-ACES and a sample of influential cognitive architectures. *Note*: By integrating the complementary schools of mathematical (left half of the figure) and symbolic (right half) models, QN-MHP supports both precise mathematical analysis and real-time generation of behavior, thus capitalizing on the strengths and overcoming the weaknesses of either mathematical or symbolic modeling alone.

QN-ACTR



tk
Vision Module
Visual Buffers
Visual-location Buffers
Matching
Selection
Execution
Goal Buffer
Goal Module
Imaginal Buffer
Imaginal Module
Aural Buffers
Aural-location Buffers
Audio Module
Declarative Buffer
Declarative Module
Vocal Buffers
Manual Buffers
Preparation
Initiation
Execution
Perceptual Subnetowrk
Cognitive Subnetwork
Motor Subnetwork

QN-MHP-BE or QN-MHP



QN-ACT-R: python

- Uses **pyactr codes**, PLUS **QN codes**, for:

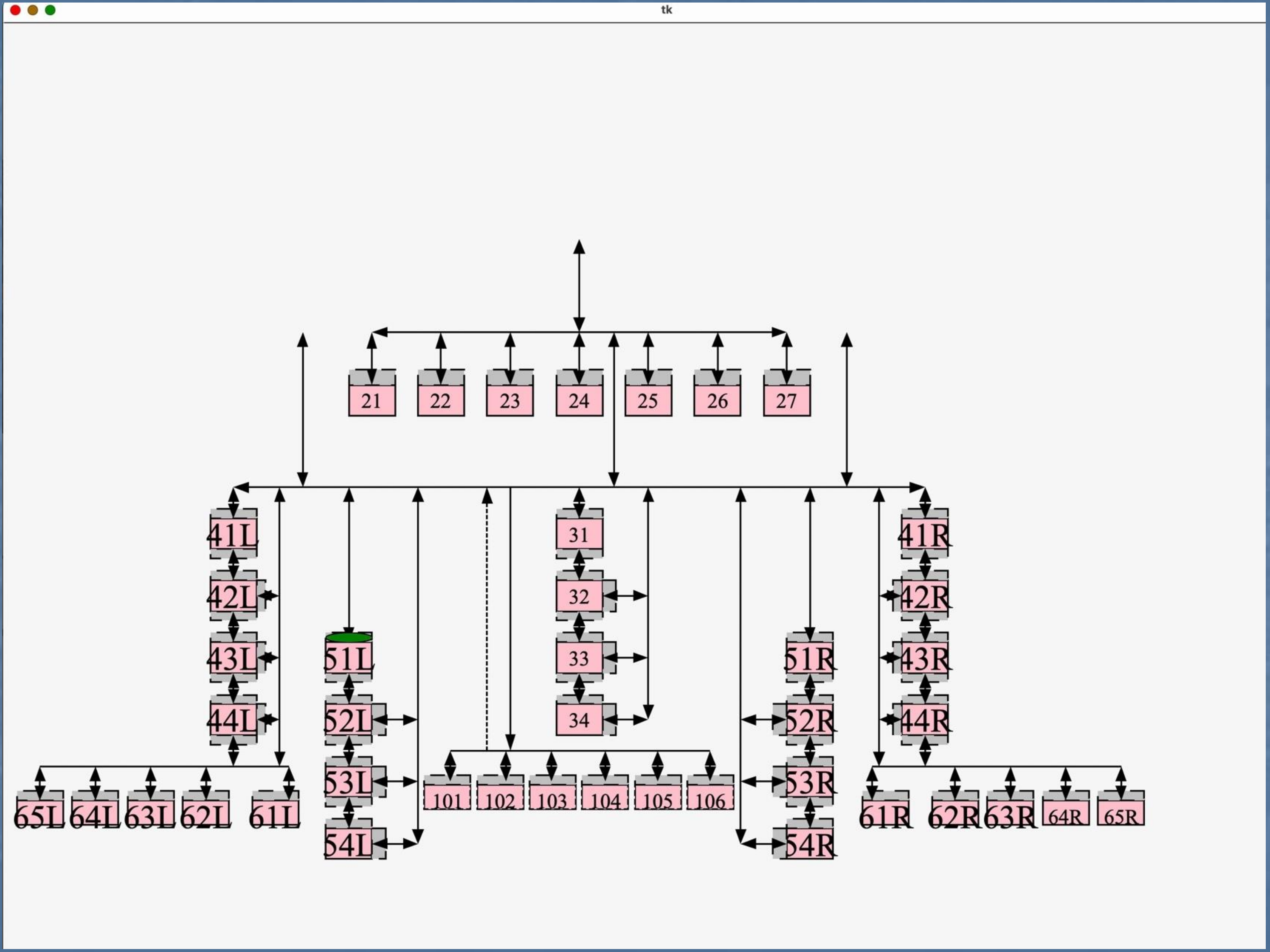
1. **QN-visualization**

2. cross-tutorial-unit **QN-ACT-R instructional work**

3. **QN-ACT-R multi-tasking research work**

Software Demo

QN-BDS



tk
21
22
23
24
25
26
27
41L
42L
43L
44L
31
32
33
34
51L
52L
53L
54L
51R
52R
53R
54R
41R
42R
43R
44R
65L
64L
63L
62L
61L
101
102
103
104
105
106
61R
62R
63R
64R
65R

Methods-time measurement (MTM)

- Purpose
  - Predict task time
    (overcome a limitation of time study)

- How?
  - Decompose tasks into elemental motions
  - Use pre-determined timing data of the elemental motions to estimate total task time

TABLE I—REACH—R

Distance Moved Inches	Time TMU A	Time TMU B	Time TMU C or D	Time TMU E	Hand In Motion A	Hand In Motion B
¾ or less	2.0	2.0	2.0	2.0	1.6	1.6
1	2.5	2.5	3.6	2.4	2.3	2.3
2	4.0	4.0	5.9	3.8	3.5	2.7
3	5.3	5.3	7.3	5.3	4.5	3.6
4	6.1	6.4	8.4	6.8	4.9	4.3
5	6.5	7.8	9.4	7.4	5.3	5.0
6	7.0	8.6	10.1	8.0	5.7	5.7
7	7.4	9.3	10.8	8.7	6.1	6.5
8	7.9	10.1	11.5	9.3	6.5	7.2
9	8.3	10.8	12.2	9.9	6.9	7.9
10	8.7	11.5	12.9	10.5	7.3	8.6
12	9.6	12.9	14.2	11.8	8.1	10.1
14	10.5	14.4	15.6	13.0	8.9	11.5
16	11.4	15.8	17.0	14.2	9.7	12.9
18	12.3	17.2	18.4	15.5	10.5	14.4
20	13.1	18.6	19.8	16.7	11.3	15.8
22	14.0	20.1	21.2	18.0	12.1	17.3
24	14.9	21.5	22.5	19.2	12.9	18.8
26	15.8	22.9	23.9	20.4	13.7	20.2
28	16.7	24.4	25.3	21.7	14.5	21.7
30	17.5	25.8	26.7	22.9	15.3	23.2

CASE AND DESCRIPTION	
A	Reach to object in fixed location, or to object in other hand or on which other hand rests.
B	Reach to single object in location which may vary slightly from cycle to cycle.
C	Reach to object jumbled with other objects in a group so that search and select occur.
D	Reach to a very small object or where accurate grasp is required.
E	Reach to indefinite location to get hand in position for body balance or next motion or out of way.

QN-BDS (QN-MTM)

Software Demo

Table 3. Summary of anthropometric data for the *"Drag-with-finger"* operator development (n = 11).

No.	Anthropometric Dimension (unit: millimeters)	*M*	*SD*	Min	Max	Population Percentiles (male/female) 5th	50th	95th
1	Stature (S)	1743	88	1622	1857	1651/1527	1755/1629	1868/1738
2	Finger spread (FS)	146	22	111	171	unknown	unknown	unknown
3	Thumb breadth (TB)	21	2	17	24	22/19	24/21	26/23
4	Index finger breadth (IB)	16	2	14	19	18/15	20/17	23/19
5	Short thumb length (STL)	64	8	55	74	62/56	70/63	78/72
6	Long thumb length (LTL)	128	10	109	145	124/112	138/125	153/141
7	Index finger length (IL)	73	6	63	84	67/62	75/70	84/77
8	Hand length (HL)	187	14	166	210	179/163	194/178	212/195
9	Hand breadth (HB)	84	7	73	95	86/76	95/83	105/90

participants had normal or corrected-to-normal vision and were right-handed. They reported no physical issues in using touchscreen display and used touchscreen devices (e.g., smartphones and tablets) for 7.5 years. Participants were paid for their time with #15 hourly rate in cash. Table 3 summarizes anthropometric data obtained from the participants, and population percentile data extracted from Greiner (1991) to compare with the participants' data.

3.2. Apparatus

A motion tracking system (OptoTrak® Certus™; Northern Digital Inc.) with two standing position sensors (three cameras on each sensor; 3.5 m away from each other) was used to record finger movements for finger-drag gestures. One marker was attached on the center of participants' right index fingernail and it was secured with Velcro® straps across the finger, wrist, and forearm, as shown in Figure 3. A touchscreen device (iPad; 1024 × 768; 132 ppi; 9.7-inch LED-backlit glossy widescreen multi-touch display) was mounted on the table (height = 95 cm). Participants were asked to find the most comfortable standing position so they do not feel any discomfort while performing the finger-drag gesture tasks.

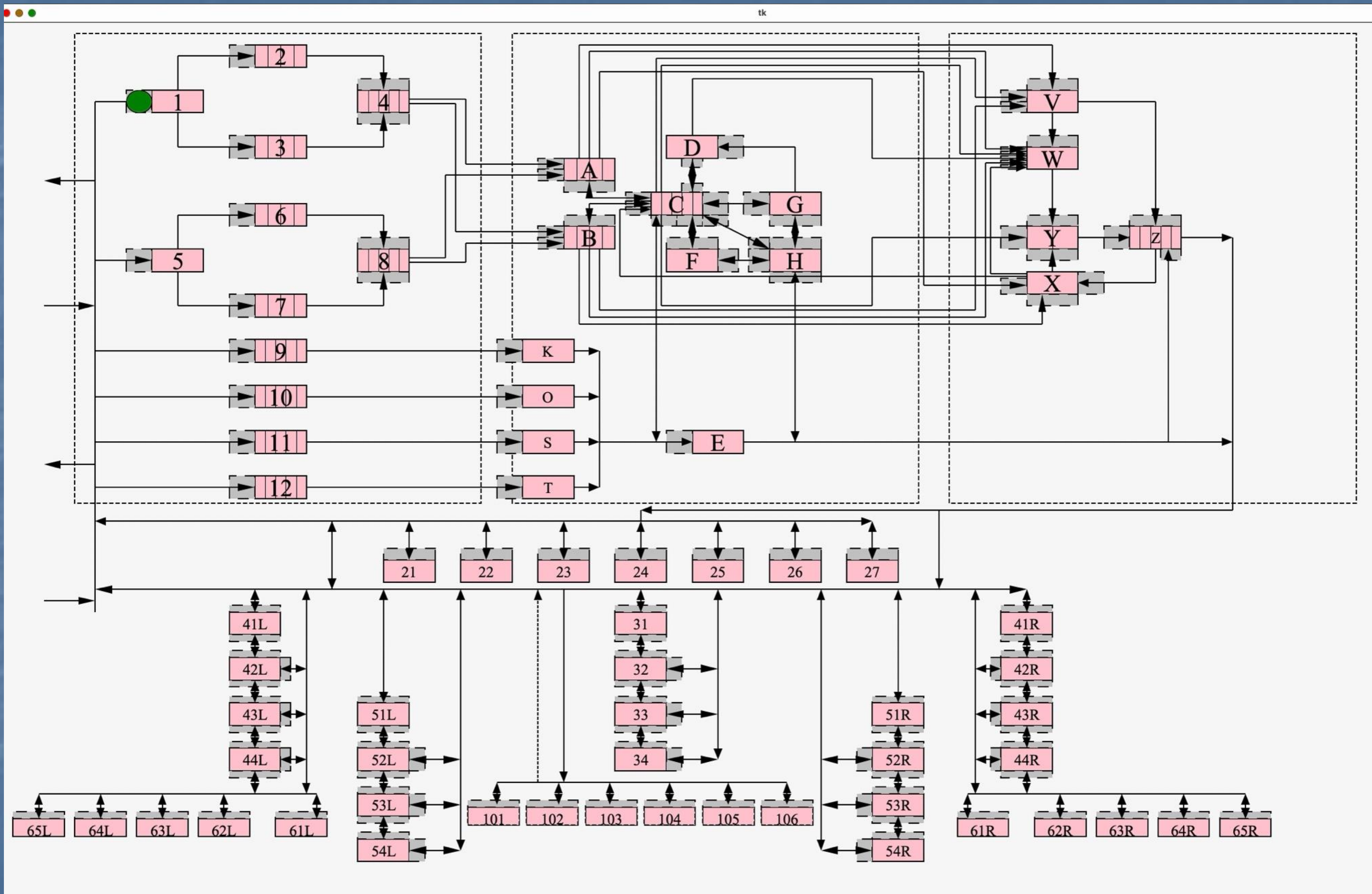
Figure 3. Experimental setup of the finger-drag gesture task.

3.3. Touchscreen gesture task and experimental design

The performance of finger-drag gestures was measured, using a touchscreen interface used in Jeong and Liu (2017a). As shown in Figure 4, nine circles (i.e., eight target circles around one center circle) were designed on a touchscreen display, but only two circles (i.e., one center circle and one of the eight target circles; colored in green – one with a hole, the other without a hole) were presented to the participants during the experiment. The distance between the center and the target circles was 40 mm, a fixed value. Participants were instructed to move their right index fingers from the center circle to one of the target circles; while the center circle was fixed and always presented, the target circle was randomly presented on the touchscreen display. The radius of both the center and target circles was identically 20 pixels (= 9 mm). Whenever the finger arrives from the air to the display surface and leaves from the display surface to the air, a 20-pixel-radius black circle was shown on the display, as a visual feedback. Only when the Euclidean distance between the center of the black and the green circles equals to or less than 20 pixels (also called a match of the black and green circles), it was defined as a success. The participants were asked to press "start" button on the center of the screen (then the button disappeared immediately), and then to complete the dragging task with two successes on the center and target circle matches (i.e., initial and final matches). In the current study, only the data of the success task trials were used and analyzed. In other words, we did not model the accuracy of the drag-gesture's performance. Instead, we used and analyzed time data, only for the success task trials.

A within-subject factorial design was used in this study. The two independent variables (including a subject variable) manipulated in this experiment were 8 different angular directions (i.e., 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°) and the participants' 9 anthropometric parameters (i.e., S, FS, HB, IB, STL, LTL, IL, HL, and HB). Each of the 11 participants conducted three replications of the drag gesture to each angular direction. Each participant performed finger-drag gestures in two sessions: (1) horizontal (0° and 180°)/vertical (90° and 270°) and (2) diagonal direction (45°, 135°, 225°, and

QN-MBS



tk
1
2
3
4
5
6
7
8
9
10
11
12
A
B
C
D
F
G
H
E
K
O
S
T
V
W
Y
Z
X
21
22
23
24
25
26
27
41L
42L
43L
44L
51L
52L
53L
54L
31
32
33
34
101
102
103
104
105
106
51R
52R
53R
54R
41R
42R
43R
44R
65L
64L
63L
62L
61L
61R
62R
63R
64R
65R

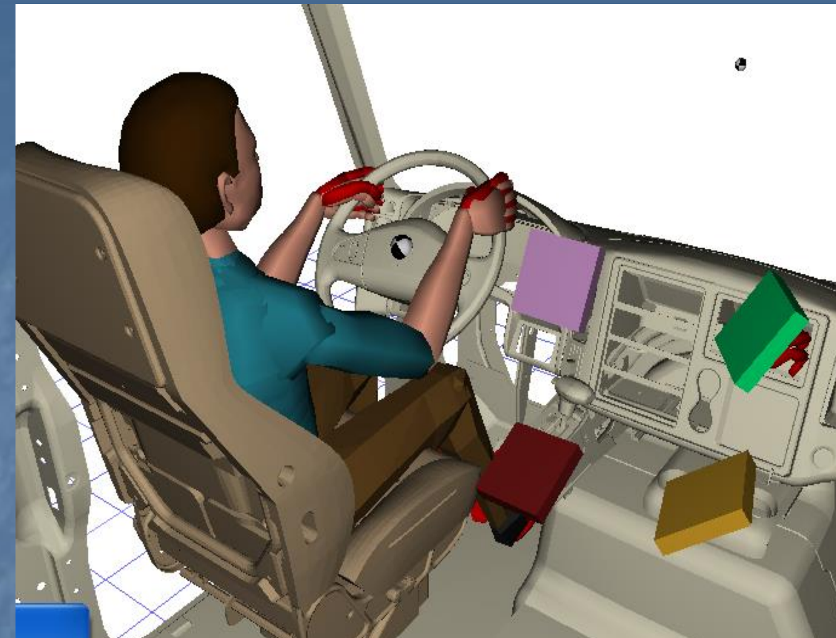
Integration of Cognitive Model with Biomechanical model
(QN and HUMOSIM-Jack)

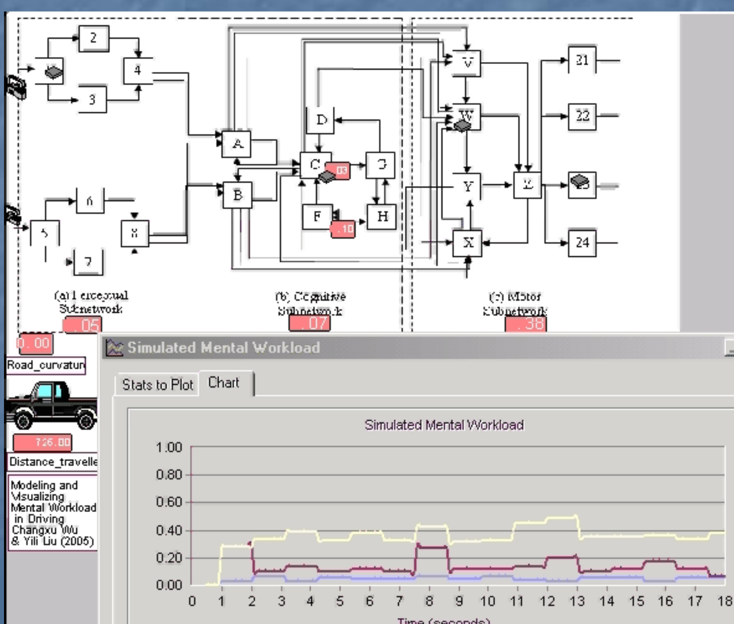
Helen Fuller, Matt Reed, Yili Liu

Supported by Automotive Research Center

Integrated Cognitive-Physical Model of Driving

- Task appears on screen
- Press button to start task
- Perform visual search for matching icons
- Press button to complete task





Near High

Far High

Near Low

Far Low

QN-Neural

NeuroImage 59 (2012) 109–116

Contents lists available at ScienceDirect

NeuroImage

journal homepage: www.elsevier.com/locate/ynimg

Computational neuroergonomics

Yili Liu [a,*], Changxu Wu [b], Marc G. Berman [c]

[a] Department of Industrial and Operations Engineering, University of Michigan, USA
[b] Department of Industrial and Systems Engineering, State University of New York at Buffalo, USA
[c] Department of Psychology and Cognitive Neuroscience Lab, University of Michigan, USA

ARTICLE INFO

Article history:
Received 20 February 2011
Revised 4 May 2011
Accepted 8 May 2011
Available online 18 May 2011

ABSTRACT

Neuroergonomics merges neuroscience and ergonomics for the study of brain and behavior in natural and naturalistic settings. Together with the rapid development of neuroergonomics concepts, technologies, and related data, there is an urgent need to develop computational models of neuroergonomics that can help integrate and interpret empirical findings and make predictions for scientific research and practical application. This article discusses the relationship between computational neuroscience and computational neuroergonomics, and describes a queuing network based computational neuroergonomic architecture and its applications. These discussions illustrate the mission and challenges of computational neuroergonomics and future research needs.

Introduction

Neuroergonomics merges neuroscience and ergonomics for the study of brain and behavior at work in relation to every day environments, technologies and settings in the real world (Parasuraman and Rizzo, 2008; Parasuraman and Wilson, 2008). As a rapidly developing field, it is accumulating a large amount of empirical data, developing a fast growing body of concepts and technologies, and opening up new frontiers of theoretical inquiries and practical applications. As in all fields of science and applications, researchers also see the importance and necessity to develop computational models of neuroergonomics in addition to data collection, concept formation, and technology development.

Computational modeling can help researchers and practitioners organize and summarize empirical data, provide unique perspectives to examine and interpret the data, guide and generate hypotheses for new experiments to obtain data in a systematic fashion, and make extrapolations and predictions to situations in which empirical data collection is not feasible or undesirable. Scientifically, computational modeling help advance the understanding of the brain and behavior; practically, it may help guide system design by comparing different design alternatives and predicting human behavior in operational contexts that cannot be easily covered by experimental or simulation studies.

One may ask "don't we have many computational models in neuroscience?" and "can't we simply use them in neuroergonomics?" The answer is that we do have many outstanding computational models in neuroscience, which can be used to inform and inspire computational neuroergonomics models; but in the same sense that neuroscience is not synonymous with neuroergonomics, computational neuroscience is not synonymous with computational neuroergonomics. In the same sense that neuroscience provides a rich soil for neuroergonomics to grow and also benefit from it, computational neuroscience models will support and hopefully benefit from computational neuroergonomics models.

In the following section (A brief summary of some related computational neuroscience models), we first summarize some of the major existing models of computational neuroscience, with a focus on models of neuroimaging data. This summary shows the strengths of these models and their limitations for direct application in neuroergonomics. In the section "A queuing network architecture of computational neuroergonomics", we discuss the unique challenges faced by computational neuroergonomics models, and describe one example of a computational neuroergonomic architecture called the queuing network architecture. In Neuroergonomic applications of the QN architecture section, some example applications of the queuing network architecture in neuroergonomics are described. The last section concludes this article by emphasizing the unique mission of computational neuroergonomics and its relation to computational neuroscience.

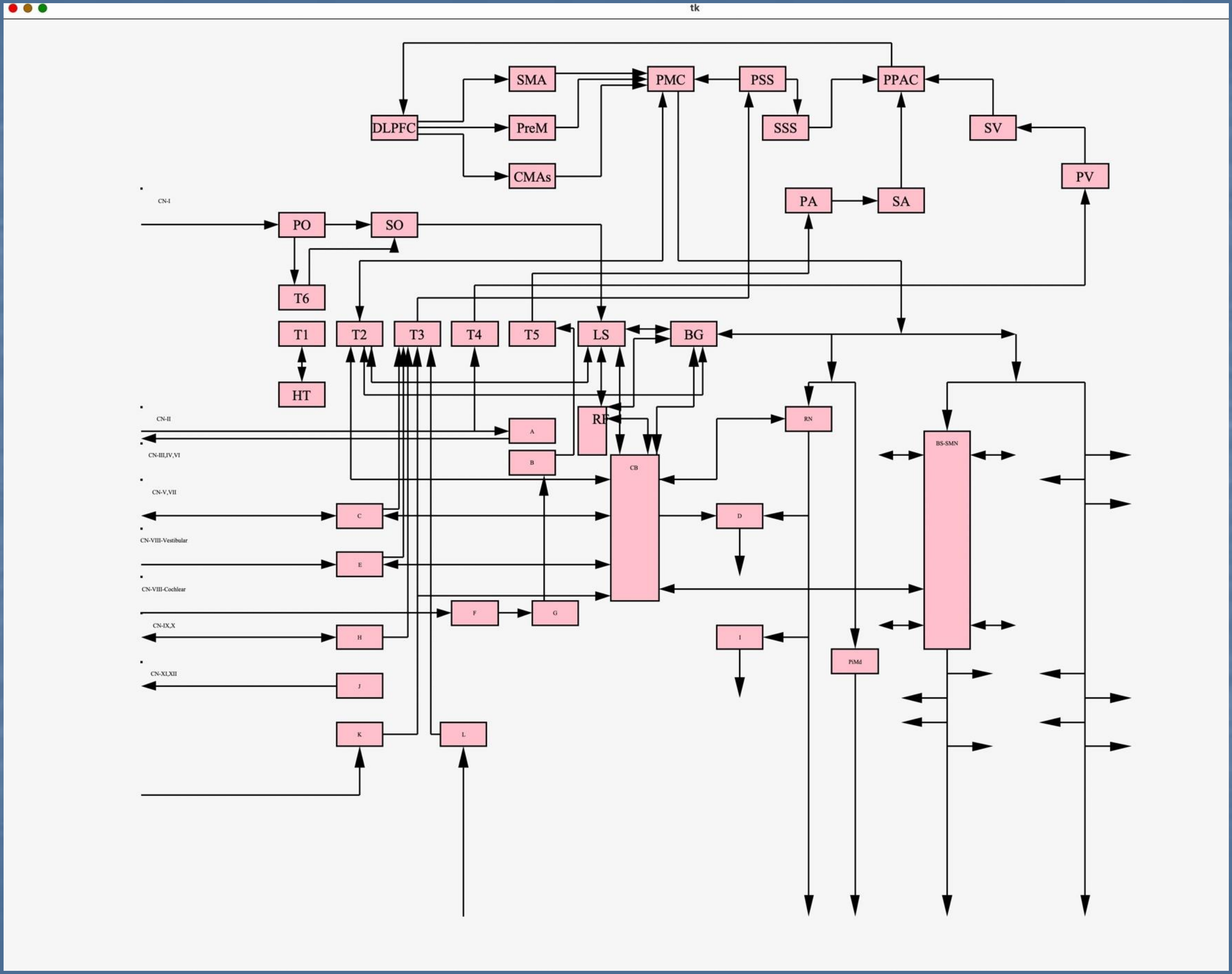
A brief summary of some related computational neuroscience models

Computational neuroscience is an extremely active field of research, and many outstanding computational models in neuroscience have been developed, from neural networks models, to statistical regression models, to causal network models, among others. It is neither feasible nor the intention of the present paper to offer a

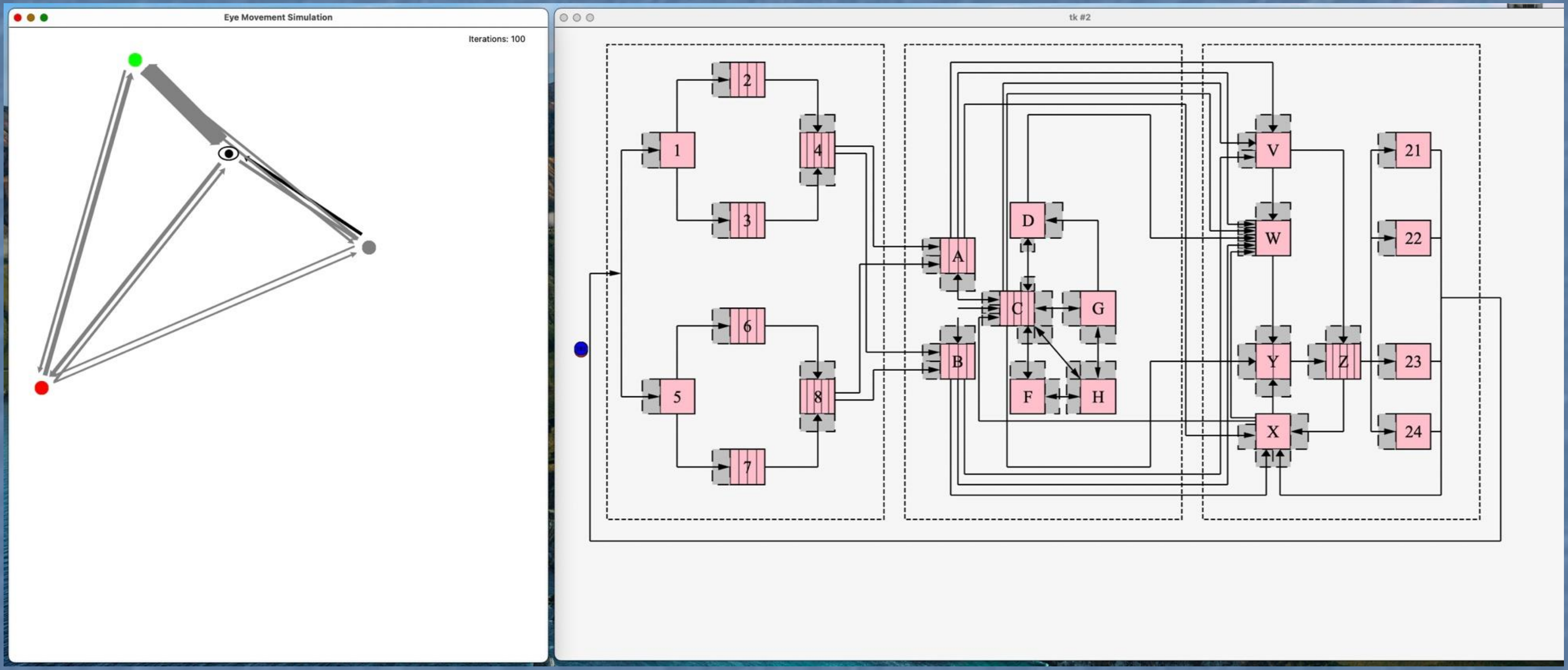
* Corresponding author at: Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109-2117, USA.
E-mail address: yililiu@umich.edu (Y. Liu).

1053-8119/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.neuroimage.2011.05.027

QN-NES


tk
DLPFC
SMA
PreM
CMAs
PMC
PSS
SSS
PPAC
SV
PV
PA
SA
CN-I
PO
SO
T6
T1
T2
T3
T4
T5
LS
BG
HT
CN-II
CN-III,IV,VI
CN-V,VII
CN-VIII-Vestibular
CN-VIII-Cochlear
CN-IX,X
CN-XI,XII
A
B
C
D
E
F
G
H
I
J
K
L
RF
CB
RN
BS-SMN

QN-SEEV



Eye Movement Simulation
Iterations: 100
tk #2
1
2
3
4
5
6
7
8
A
B
C
D
F
G
H
V
W
X
Y
Z
21
22
23
24

QN-SEEV

Software Demo

QN-Reinforcement Learning-Eye Movement

Modeling the Influences of Cyclic Top-Down and Bottom-Up Processes for Reinforcement Learning in Eye Movements

Ji Hyoun Lim and Yili Liu, *Member, IEEE*

***Abstract*—Understanding and reproducing complex human oculomotor behaviors using computational models is a challenging task. In this paper, two studies are presented, which focus on the development and evaluation of a computational model to show the influences of cyclic top-down and bottom-up processes on eye movements. To explain these processes, reinforcement learning was used to control eye movements. The first study showed that, in a picture-viewing task, different policies obtained from different picture-viewing conditions produced different types of eye movement patterns. In another visual search task, the second study illustrated that feedback information from each saccadic eye movement could be used to update the model's eye movement policy, generating different patterns in the following saccade. These two studies demonstrate the value of an integrated reinforcement learning model in explaining both top-down and bottom-up processes of eye movements within one computational model.**

***Index Terms*—Cognitive model, eye movement, queueing network model, reinforcement learning.**

I. Introduction

EYE movements are essential to acquire information in visual interfaces and represent one of the most prevalent activities in human–machine systems. Eye movements have been studied to understand covert and overt attention [1]–[4], spatial attention [5], [6], pattern recognition [7], [8], and influences of cognitive processes on oculomotor behavior [9]–[11] and to also improve machine vision [12]. A major consensus of these studies is that eye movements are influenced by both the perceiver's expectations (called the top-down process) and the characteristics of the visual environment (called the bottom-up process).

As Neisser [13] proposed, perception is a cyclic process in which a mental model directs the eyes to explore and sample the environment, and the information obtained from the environment is used to modify the mental model. The top-down process is driven by the mental model, while the bottom-up process is driven by the environment. Forty years ago, Yarbus [11] showed how human eye movements are influenced by top-down cognitive processes. Yarbus' experiment was recently revisited [7], [8], and the studies supported the original findings. While Yarbus' experiment demonstrated the influence of top-down processing on eye movement patterns, the influence of bottom-up processing has been shown through studies on visual saccades, a much more basic type of eye movement. Findlay's extensive studies on saccades [14]–[16] and other researchers' related studies on the distractor-ratio effect [17], [18] have shown the influence of bottom-up processes on saccadic eye movements. The distractor-ratio effect is observed in a visual search task involving two types of distractors in which, even though the total number of items in a display remains the same, visual search performance changes with the ratio between the two types of distractors.

A single saccade is the basic component of eye movement patterns. Spatial attention is assumed to play a major role in the planning and the execution of saccades, and so, the relationship between spatial attention and saccadic eye movement has been studied in depth over the last few decades (for an overview, see [15]). There are several conceptual models of spatial attention, including the spotlight theory [3], the zoom lens theory [1], and the selection theory of attention [19], [20]. The theories explaining the relationship between spatial attention, or, more specifically, covert attention, and eye movements are the premotor theory of attention [3] and the sequential attentional model [21]. Studies on the neurophysiology of attention have shown strong support for a close coupling between covert attention and saccade generation in that both arise from the same basic neural processes [22], [23]. The sequential attentional model and the premotor theory both assume a close link between covert attention and saccades. The sequential attentional model assumes that a saccade is driven by a covert attention shift, while the premotor theory assumes that a covert attention shift is a by-product of the action of the oculomotor system.

In the field of human factors, eye transition patterns have been studied in various task domains [24]. In the statistical models of eye transition, the open-loop versus closed-loop control is one issue of debate [24]. The open-loop control strategy assumes repetitive information gathering independent of the previous eye fixation. This strategy emphasizes the top-down influence on eye movements. On the other hand, the closed-loop strategy assumes that an eye movement is dependent on information gathered from the previous fixation. The

Manuscript received November 17, 2006; revised November 28, 2007 and June 27, 2008. Current version published June 19, 2009. This work was supported by the National Science Foundation under Grant 0308000. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation. This paper was recommended by Associate Editor Y. Xiao.

J. H. Lim is with the Design Team, Mobile Communication Division, Samsung Electronics, Seoul 100-742, Korea (e-mail: smilelim@gmail.com).

Y. Liu is with the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: yililiu@umich.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2009.2018635

QN-Reinforcement Learning-Eye Movement

after each selected action (a_t) in a given state (s_t) provides information about how good the action was, but it says nothing about whether the action was correct or incorrect. Therefore, a reward function has to be defined based on the goal and properties of the information given by the environment. In eye movement modeling, the reward is determined by whether the perceived information delivers what the agent is looking for or not.

The third feature of reinforcement learning is the Markov property of the learning process. The Markov property states that the probability distribution of the future states of the process $(\Pr\{s_{t+1} = s', r_{t+1} = r | s_t, a_t\})$ depends only upon the current state, and is conditionally independent of the past states (the path of the process) given the current state. As long as the learning process has the Markov property, the expected value of reward $(\mathrm{R}^a_{ss'} = \Pr\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\})$ can be calculated based on the transition matrix of probabilities between the states $(\mathrm{P}^a_{ss'} = \Pr\{s_{t+1} = s' | s_t = s, a_t = a\})$ and the reward function.

A policy (π) defines how the learning agent selects an action at a given state. The main interest in reinforcement learning is to find the policy that delivers the maximum reward. There are two major issues in finding the policy that delivers the maximum reward: 1) how to evaluate the candidate policies and 2) how to update policies. One way to evaluate policies is to assess the value of a state. The other way is to evaluate the value of an action. In this study, we used the value function $Q(s, a)$ to evaluate actions. The definition of $Q(s, a)$ is provided in the next section.

III. Eye Movements and Reinforcement Learning

Reichle and Laurent [31] used reinforcement learning to understand eye movements in reading by assuming an artificial reader being capable of learning to control its eye movements. Their model proposed nine variables defining the states of the reading agent, and a reward was given for each word identified. Sprague and Ballard [32] examined temporal scheduling of human eye movements based on reinforcement learning. The task in their study was to find multiple targets in a given environment. The model tried to schedule the temporal order of targets to be looked at to maximize the total rewards. The reward for each target was assigned by the researchers as varying amounts with a predefined unit. Their study focused on strategic scheduling in eye movements rather than explaining the underlying processes in human eye movements.

Our approach to modeling eye movements was to consider each eye movement as a Markov decision process and then find an underlying policy using the reinforcement learning method. The Markov process assumption is essential for finding the optimal policy (π^*). The goal of our paper, however, was not to find the optimal policy for eye movements but to focus on the differences in policies (π) under different visual task conditions. Therefore, the Markov process assumption was not crucial for this paper but helped to map eye movements in a reinforcement learning problem.

A Markov decision process has four attributes: S is the state space, A is the action space, $\mathrm{P}^a_{ss'}$ is the transition matrix that

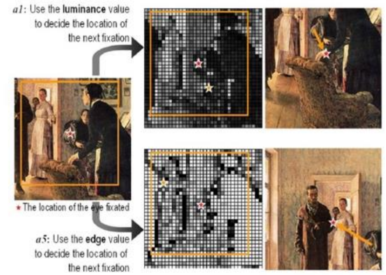


Fig. 1. Location selected for the next eye fixation depends on the action chosen.

indicates the probability of arriving in state s' when action a is taken in state s, and $\mathrm{R}^a_{ss'}$ is the expected reward value. To select an action in a given state, the optimal value function $Q(s, a)$ is used in this study. The optimal value function $Q(s, a)$ represents the expected discounted return if action a is taken in state s and the optimal policy is followed thereafter. Among many algorithms [30] for updating the value of $Q(s, a)$, the online Q-learning update rule was used to select a feature for a saccadic eye movement

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a)\right].$$

Here, α is a learning rate parameter, and γ is a discounting factor of future reward. In this study, the initial value of the learning rate parameter α_0 was 0.3, and it decreased by following the rule

$$\alpha_n = 0.99\, \alpha_{n-1}.$$

The discounting factor of future reward γ was set at 0.5 to consider immediate reward, as well as future reward, through the learning process. The value of the discounting factor did not change so as to reduce computational complexity.

To map a reinforcement learning model to eye movements, the major components in reinforcement learning—agent, environment, states, action, reward, and policy—were defined as follows.

The ***Agent*** was defined as an eye with attended and unattended visual zones. In Fig. 1, the star indicates the attended visual zone, and the large square indicates the unattended visual zone. To simplify the calculation process during computational simulation, the square shape was used to represent the visual zones. The location of eye fixation ($x \in E$, where x is a 2-D vector) was determined as the location of the center of the unattended visual zone.

The ***Environment*** (E) was the visual stimulus (or the picture) shown to the agent to complete a given task. In Yarbus' picture-viewing task, the environment was I. E. Repin's painting, and in Findlay's visual search task, the 16 items in two concentric rings were presented as the environment.

QN-Reinforcement Learning-Eye Movement

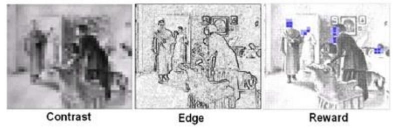


Fig. 3. Input data arrays of luminance, edge, and reward (give the ages of the people) values for I. E. Repin's painting "They Did Not Expect Him" (1884) used in the picture-viewing task.

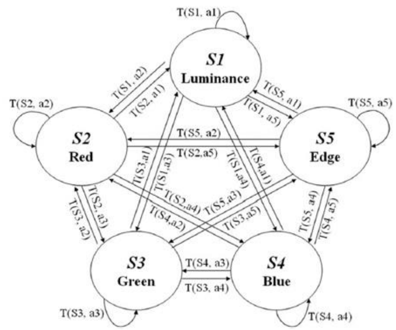


Fig. 4. Action space with 25 actions.

the location x on the original picture whose size is 0.5° of visual angle (estimated from the Yarbus' study). Fig. 3 shows the luminance and edge values used in this simulation study. The cell of the ith column and the jth row on the left array in Fig. 3 represents the amount of luminance value for the location $x(i, j)$ on the original picture. The range of values was from 0 to 255.

The expected value of ***Reward*** $\mathrm{R}^{a}_{ss'}$ was assumed to be the amount of information that the agent could retrieve at a given location x by taking the action a from the state s. In this case, the amount of information for each location x was assigned as High, Low, and None, depending on the question to be answered. For example, under the "give the ages of the people" condition shown in Fig. 3, the rewards were assumed to be distributed mainly at the faces in the picture, whereas under the "what family had been doing" condition, the rewards were assumed to be distributed mainly at the bodies of the people.

Fig. 4 shows the action space $A(s_t)$. The five states are the five alternative features (luminance, red, blue, green, and edge) currently used, and actions are staying in the current state or moving to one of the other states

$$\text{Reward per saccade} = \frac{\sum_{i=0}^{t} R_i}{t}$$

$$R_t = R^{a_t}_{s_t s_{t+1}}.$$

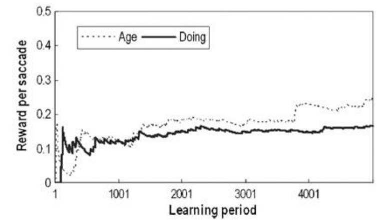


Fig. 5. Reward per saccade during the 5000 learning periods.

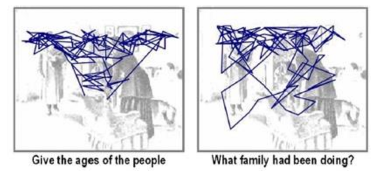


Fig. 6. Simulated eye movement patterns of 300 saccades with two different sets of rewards after 5000 learning trials.

Using 5000 randomly selected actions, the value of each $Q(s, a)$ was updated. Fig. 5 shows the changes in the amount of reward per saccade during the learning period. After the learning period, the amount of reward per saccade appeared to be stabilized, as well as the values of $Q(s, a)$ for all s's and a's. When the value of the reward per saccade reaches a plateau, it can be interpreted as the model starting to generate a certain pattern in eye movements.

As explained earlier, two different reward sets were used for the "Give the ages of the people" condition and the "What the family had been doing" condition. The two $F(s, a)$ matrices for these conditions are

$$F(s,a)_{\mathrm{Age}} = \begin{bmatrix} 0.06 & 0.06 & 0.06 & 0.77 & 0.06 \\ 0.19 & 0.19 & 0.19 & 0.19 & 0.23 \\ 0.02 & 0.05 & 0.02 & 0.87 & 0.04 \\ 0.17 & 0.17 & 0.17 & 0.17 & 0.32 \\ 0.24 & 0.19 & 0.19 & 0.19 & 0.20 \end{bmatrix}$$

$$F(s,a)_{\mathrm{Doing}} = \begin{bmatrix} 0.16 & 0.16 & 0.16 & 0.16 & 0.36 \\ 0.04 & 0.04 & 0.76 & 0.04 & 0.11 \\ 0.18 & 0.18 & 0.18 & 0.22 & 0.26 \\ 0.15 & 0.15 & 0.40 & 0.15 & 0.16 \\ 0.25 & 0.10 & 0.33 & 0.12 & 0.20 \end{bmatrix}.$$

Using the previous two policies, the viewing tasks were simulated. Fig. 6 shows the simulated eye movements of 300 saccades. The simulated eye movement patterns showed difference between the two different viewing conditions. The only difference in modeling the two conditions—"Give the ages" and "What the family had been doing"—was the *location of rewards*. All the other parameters, input data, and procedural rules for eye movements were exactly the same for both cases.

QN-Reinforcement Learning-Eye Movement

IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 11, NO. 4, DECEMBER 2010 765

Investigation of Driver Performance With Night-Vision and Pedestrian-Detection Systems—Part 2: Queuing Network Human Performance Modeling

Ji Hyoun Lim, Yili Liu, *Member, IEEE*, and Omer Tsimhoni, *Member, IEEE*

***Abstract*—This paper introduces a queueing network-based computational model to explain driver performance in a pedestrian-detection task assisted with night-vision-enhancement systems. The computational cognitive model simulated the pedestrian-detection task using images displayed by two night-vision systems as input stimuli. The system equipped with a far-infrared (FIR) sensor generated less-cluttered images than the system equipped with a near-infrared (NIR) sensor. Using a reinforcement learning process, the model developed eye-movement strategies for each night-vision system. The differences in eye-movement strategies generated different eye-movement behaviors, in accord with the empirical findings.**

***Index Terms*—Cognitive model, human performance modeling, night vision, pedestrian detection, queueing network.**

I. Queueing Network Model of Pedestrian Detection and Driving

DRIVER performance is a critical factor in examining the effectiveness and efficiency of an intelligent transportation system (ITS) [1], [2]. Computational models of driver performance can help study driver behavior and assist system design. This study introduces an example of using a computational model to assess driver performance using ITSs (two night-vision-enhancement systems designed to assist in pedestrian-detection tasks). In this study, the near-infrared (NIR) night-vision-enhancement system and the far-infrared (FIR) system were investigated. The NIR system actively illuminates a scene in the NIR spectrum and captures the reflected radiation, whereas the FIR system generates images by passively detecting thermal emissions from objects in a scene of interest.

Manuscript received October 23, 2008; revised April 18, 2010; accepted April 27, 2010. Date of publication June 7, 2010; date of current version December 3, 2010. The Associate Editor for this paper was N. B. Sarter.

J. H. Lim was with the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109 USA. He is now with the Mobile Communication Division, Samsung Electronics, Seoul, 137-857 Korea (e-mail: smilelim@gmail.com).

Y. Liu is with the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109-2117 USA.

O. Tsimhoni was with the University of Michigan Transportation Research Institute and Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109 USA. He is now with the General Motors Advanced Technical Center-Israel, Herzliya 46725, Israel (e-mail: omer.tsimhoni@gm.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2010.2049844

The two night-vision systems generate distinctively different images of the same scene, which may lead to different glance behaviors, such as different glance frequencies and search times to detect pedestrians during night driving. In this study, pedestrian detection with a concurrent driving task was simulated with a computational cognitive model, based on the queueing network architecture, to investigate the different glance behaviors associated with the different night vision systems.

As shown in Fig. 1, the cognitive agent (driver) processes information from the images displayed by the night-vision-enhancement systems and then generates glance behaviors (scans the scene for pedestrians). The relationship between the characteristics of a visual scene (input stimulus) and the glance behaviors has been examined in empirical studies [3]–[6]. The computational modeling approach allows us to investigate the possible underlying cognitive mechanisms of the glance behaviors.

Along the line of research on developing comprehensive human performance models with a unifying cognitive architecture, we have been making steady progress in developing a queuing network (QN) architecture for human performance modeling. Mathematical models based on QNs have successfully integrated a large number of models in response time [7] and in multitask performance [8] as special cases of QNs. A computational model based on the QN mental architecture called the queueing network—model human processor (QN-MHP) [9] has been developed to simulate and generate human performance. This architecture represents a human psychological system in the form of a queueing network with multiple servers that represent functional units in a human brain. Entities represent pieces of information to be processed by the servers. An entity travels on routes, which represent the flow of information in the system. The QN-MHP has successfully been applied to model various task domains including simple and choice reaction tasks [10]–[12], visual search tasks [10], [13], driving [14], [15], driving with a concurrent map reading task [7], and driver workload [16]. For a detailed description of the QN-MHP and its applications methodology, see [9].

Adopting the QN-MHP methodology, a QN menu search model was developed in our earlier study [13], which was significantly extended in the present study to establish a QN model of pedestrian detection. There are nine effective servers for the menu search model used in the previous study [13], which were kept in this pedestrian-detection model. A

QN-Reinforcement Learning-Eye Movement

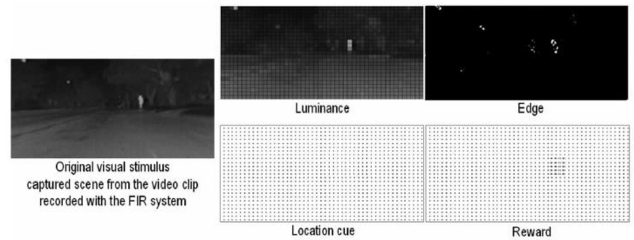


Fig. 3. Example of the input data for the pedestrian-detection task.

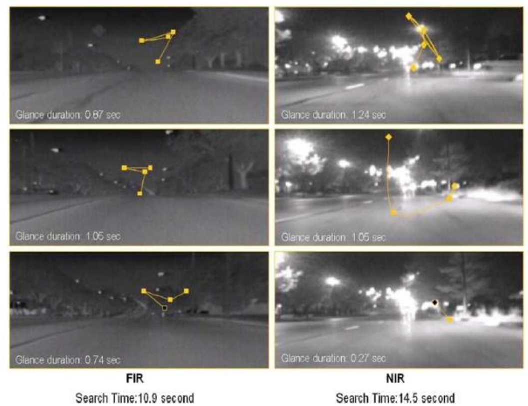


Fig. 4. Example of eye movements for the last three glances before response.

For reward ($f = 4$), a constant positive number was assigned to the location where the pedestrian appeared and zero to the rest. An example of the input data sets is shown in Fig. 3.

$\boldsymbol{InputSimulus_{x,y,f,t}}$

where

x horizontal location ($x = 1, 2, \ldots, 48$);
y vertical location ($y = 1, 2, \ldots, 25$);
f feature ($f = 1$ for the luminance, 2 for the edge, 3 for the location cue, and 4 for the reward);
t time ($t = 1, 2, \ldots, 20$).

When the eyes were fixated on a specific location, the two types of visual stimuli were encoded based on the data array $\boldsymbol{InputSimulus_{x,y,f,t}}$. One stimulus, representing attended visual information, carries information on all the three features (luminance, edge, and location cue). The other type of stimulus, representing preattended visual information, carries only information on one feature, which is currently selected as the hunt feature by the cognitive system.

The sets of visual stimuli were the scenes displayed by two different night-vision-enhancement systems—NIR and FIR—to assist pedestrian detection during night driving. Due to the different sensing mechanisms, the two night-vision systems generate distinctively different scenes, as shown in Fig. 4.

3) Eye-Movement Strategy for the Pedestrian Detection: To model the pedestrian-detection task, the hunt-feature rule and a reinforcement learning process were used as the procedure rule for eye movements. The hunt-feature rule explained the procedure of the varying length in a saccade phenomenon based on the attentional spotlight theory [21] and the feature synthesis theory [22]. The reinforcement learning process allowed the cognitive system to select a visual cue as a hunt feature based on the computed values of selecting the features.

Lim and Liu [23], [24] showed that eye-movement patterns changed according to the combinations of the probabilities of selecting each feature as a hunt feature. Therefore, the reinforcement learning process was integrated into the QN model to

QN-Control (Classical)

1812 IEEE TRANSACTIONS ON INTELLIGENT

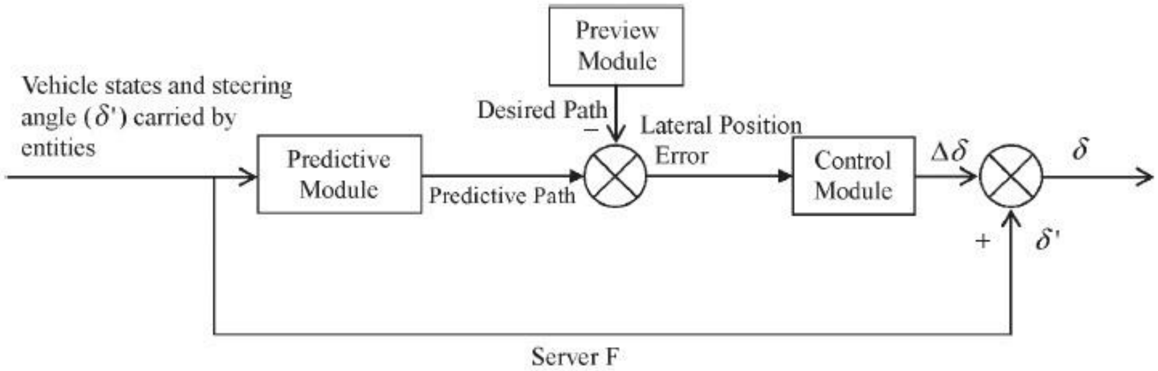


Fig. 2. Schematic of driver lateral control implemented in Server F.

The preview module previews the desired path for a preview interval to obtain information of the desired path. This paper

QN-Control (Classical)

BI *et al.*: DEVELOPMENT OF DRIVER LATERAL CONTROL MODEL BY INTEGRATING NEUROMUSCULAR DYNAMICS 2481

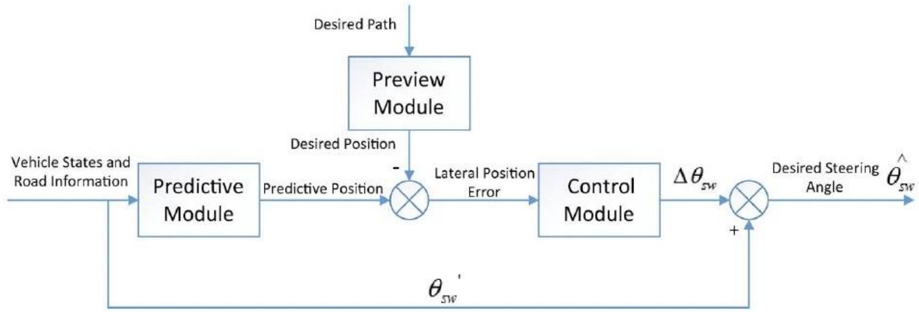


Fig. 3. Block diagram of driver preview model implemented in Server F.

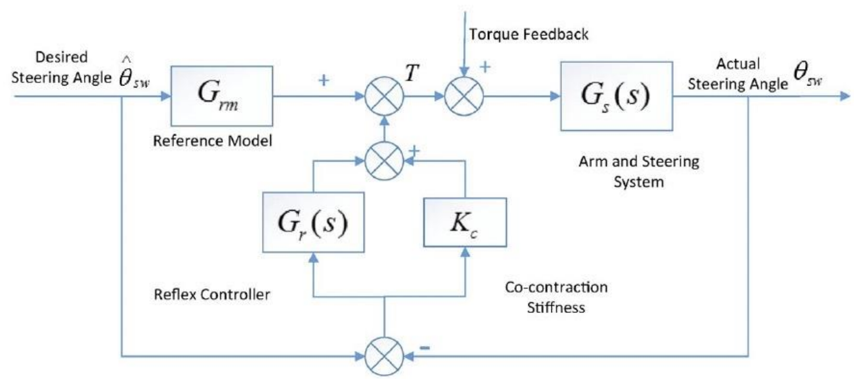


Fig. 4. The structure of the neuromuscular system.

QN-Control (Classical)

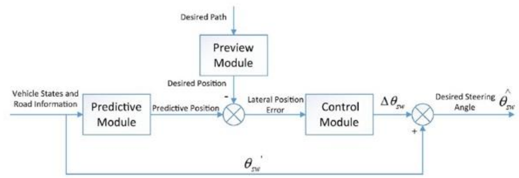


Fig. 3. Block diagram of driver preview model implemented in Server F.

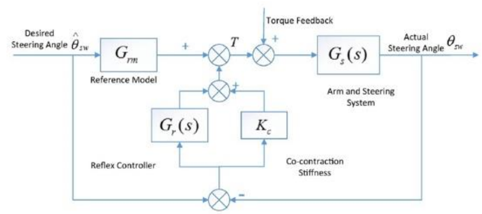


Fig. 4. The structure of the neuromuscular system.

As shown in Fig. 2, entities carrying the inputs of the entire model first enter the visual perceptual subnetwork (Servers 1 (visual input) → 2/3 (visual recognition/visual location) → 4 (perceptual integrator)). Via Server 4 (Perceptual-Integrator server), the entities are routed to the cognitive subnetwork, including Servers A (visuospatial sketchpad), C (central executor), and F (complex cognitive function), where the desired steering angle is computed according to the preview model, as described in Fig. 3. Then entities carrying the desired steering angle $\hat{\theta}_{sw}$ travel to the motor subnetwork via Server C, where the desired steering angle carried by entities is transported into the neuromuscular system via Server Z.

As shown in Fig. 3, the driver preview model implemented at Server F of the QN architecture contains three main modules: preview module, predictive module, and control module. The preview module previews the desired path for a preview time to obtain the information of the desired path (note that the desired path was predetermined). The predictive module predicts the vehicle response within the preview time by using an internal model (i.e., 3 degree-of-freedom (DOF) vehicle dynamics model, including longitudinal, lateral, and yaw movement).

The control module computes the desired control input (i.e., the increment of steering angle $\Delta\theta_{sw}$) to make a vehicle track the desired path. The calculation proceeds as follows [1]. The desired acceleration a_y (assuming that it is a constant in the preview time) is first calculated by

$$a_y = \frac{2 \times (\Delta E - v \times t_p)}{t_p^2}, \tag{1}$$

where ΔE is the error between the desired lateral position obtained with the desired path and predictive lateral position computed with the internal vehicle dynamics model, v is the current velocity, and t_p is the preview time.

The changed steering angle $\Delta\theta_{sw}$ is then calculated with a proportional derivative (PD) controller of acceleration:

$$\Delta\theta_{sw} = k_p \times a_y + k_d \times a_y', \tag{2}$$

where k_p and k_d are the coefficients of the PD controller, and a_y' is the first derivative of the acceleration.

Finally, a new steering angle $\hat{\theta}_{sw}$ is computed by

$$\hat{\theta}_{sw} = \Delta\theta_{sw} + \theta_{sw}', \tag{3}$$

where θ_{sw}' is the steering angle of the last cycle. More details on the QN-based driver lateral control can be found in our previous work [1].

C. Neuromuscular Dynamics

The neuromuscular system (NMS) was developed according to [6] and implemented outside of the QN architecture. The difference between the neuromuscular system in this paper and that of [6] was in the parameter values. The model was used to produce an actual steering angle given the reference input (i.e., the desired steering angle). It can reject any disturbance torque on the steering wheel, to which an external disturbance leads. The structure of NMS used is shown in Fig. 4 [6].

The proposed model includes feedforward module G_{rm}, the co-contraction stiffness K_c, the stretch reflex controller $G_r(S)$, and the arm and steering system $G_s(s)$. G_{rm} represents the angle-torque stiffness, which can provide a steering torque proportional to the desired angle $\hat{\theta}_{sw}$. K_c denotes the increased

QN-Control (Modern)

content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3342151

output discrete driving commands with limited numbers in the brain-control driving, while they send continuous driving signals in the manual-control driving; 2) to execute their driving intentions, drivers perform brain-control operation via brain signals in the brain-control driving, while they conduct limb-control operation via muscles and peripheral nerves in the manual-control driving.

Since the car-following behavior of the brain-control driving consists of the car-following decision and brain-control operation sub-behaviors, we model it by fusing the models of the two sub-behaviors. As shown in Fig. 2, we first design a car-following decision model to simulate the car-following decision of the brain-control drivers. Then, the brain-control behavior and BCI performance models are applied to simulate the brain-control operation of the users. After that, we build two brain-control car-following models (i.e., CF-BCB and CF-BCI models) by combining the car-following decision model with the brain-control behavior and BCI performance models in the QN architecture, respectively. The QN architecture, including perceptual and cognitive subnetworks, is employed to denote human information processing in performing the brain-control driving.

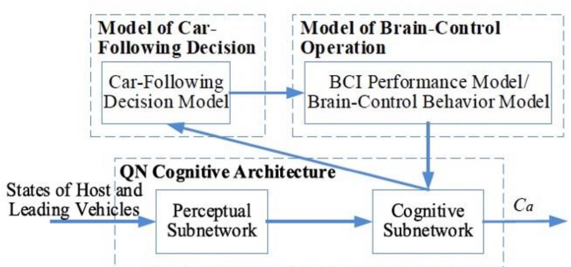


Fig. 2. Architecture of the brain-control car-following models.

MPC method, which performs well in the modeling of driving behaviors [13], [17], [18]. In the MPC algorithm, we use the longitudinal models of the host and leading vehicles as prediction models. They enable the car-following decision model to predict the future states of the two vehicles. The cost function incorporates a car-following and a smooth control objective.

1) Vehicle Model

The host vehicle is expected to imperfectly track its desired acceleration. Then, we can describe its longitudinal model with a first-order lag as

$$\tau\dot{a} + a = u \tag{1}$$

where τ denotes the time lag [33], and a and u represent the actual and desired accelerations of the host vehicle, respectively.

Integrating forward difference approximations, the dynamic model in (1) can be rewritten as

$$x(k+1) = Ax(k) + Bu(k)\,,$$
$$A = \begin{bmatrix} 1 & T & 0 \\ 0 & 1 & T \\ 0 & 0 & 1-T/\tau \end{bmatrix},\ B = \begin{bmatrix} 0 \\ 0 \\ T/\tau \end{bmatrix}. \tag{2}$$

where T denotes the sampling time and $x(k) = [s(k), v(k), a(k)]^T$ represents the state variable of the host vehicle. s and v are the absolute position and longitudinal velocity of the host vehicle, respectively. The users were allowed to control the host vehicle within a certain velocity range. The velocity constraint can be expressed as

$$v_{\min} \le v \le v_{\max} \tag{3}$$

where $v_{\min}$ and $v_{\max}$ are the limits of the longitudinal velocity.

Queuing Network Modeling of Driver EEG Signals-Based Steering Control

Luzheng Bi, Yun Lu, XinAn Fan, Jinling Lian, and Yili Liu

***Abstract*—Directly using brain signals rather than limbs to steer a vehicle may not only help disabled people to control an assistive vehicle, but also provide a complementary means of control for a wider driving community. In this paper, to simulate and predict driver performance in steering a vehicle with brain signals, we propose a driver brain-controlled steering model by combining an extended queuing network-based driver model with a brain–computer interface (BCI) performance model. Experimental results suggest that the proposed driver brain-controlled steering model has performance close to that of real drivers with good performance in brain-controlled driving. The brain-controlled steering model has potential values in helping develop a brain-controlled assistive vehicle. Furthermore, this study provides some insights into the simulation and prediction of the performance of using BCI systems to control other external devices (e.g., mobile robots).**

***Index Terms*—Assistive technology, brain-controlled vehicles, electroencephalography (EEG), queuing network modeling.**

I. Introduction

HEALTHY drivers use limbs to operate a vehicle. Driver models representing this type of driver-vehicle interactions have been widely investigated [1]. Researchers can use these models to simulate and predict driving performance and thus help reduce the time and effort on experimental design and implementation in testing the performance of driver assistance systems [2]. Furthermore, these models can be used to develop adaptive driver assistance systems to better assist driving [3], [4]. Thus, driver models are valuable in helping develop driver assistant systems for improving driving performance and safety.

Some disabled people, however, cannot drive a vehicle with limbs. To help these disabled individuals control vehicles to increase their mobility, brain-controlled vehicles (BCVs) have been proposed. A brain-controlled vehicle (BCV) is a vehicle that is directly controlled by the human "mind" through interpreting brain activity signals into motion commands rather than limbs.

BCVs can be considered as one of the applications of brain–computer interfaces (BCIs). Brain signals used to develop BCIs can be measured and recorded with methods such as functional near-infrared imaging (fNIR), recording electrical or magnetic fields, functional magnetic resonance imaging (fMRI), and positron emission tomography (PET) [5], [6]. The electrical fields can be recorded at the scalp (electroencephalography (EEG) signals), on the cortex (electrocorticographic signals), or within the cortex (neuronal action potentials). Currently, compared to other methods, EEG recording is less expensive, simpler, and non-invasive, and thus more suitable for use in practice. EEG signals have been widely applied to develop various brain-controlled systems, such as brain-controlled cursors [7], [8], brain-controlled prosthesis [9], [10], and brain-controlled wheelchairs [11]–[13]. In contrast to brain-controlled wheelchairs, BCVs are more complicated in dynamic characteristics, and they travel faster in a more complicated environment.

Recently, some researchers have started to explore how to develop a BCV with EEG signals because of the advantages of EEG recordings mentioned above. Gohring *et al.* [14] explored how to employ a commercial BCI product from Emotiv to control an intelligent vehicle. If the intelligent vehicle is about to leave the predefined free driving zone or to collide instantly, it immediately stops itself and resets the BCI system. They tested the system by requiring one participant to finish driving on a closed airfield. The experimental results indicated that it was feasible to use EEG signals in conjunction with an autonomous control system to control a vehicle. In our previous work [15], we have developed a new steady-state visually evoked potential (SSVEP) BCI, whose visual stimuli were presented on a windshield with a head-up display, and used such BCI along with the alpha rhythm of EEG to operate the simulated vehicle supported by a 14-degree of freedom (DOF) vehicle dynamics model. We tested the BCV by asking four participants to finish a driving task online. The experimental results showed the feasibility of only using EEG to continuously steer vehicles. The major difference between [15] and [14] is that we developed a novel BCI and applied it to completely and continuously steer a simulated vehicle without any stops during the whole testing process, whereas [14] applied a BCI product (without reporting the performance of the BCI) to intermittently control an intelligent vehicle, which can stop itself and reset the BCI.

Manuscript received February 25, 2016; revised August 21, 2016; accepted September 24, 2016. Date of publication September 28, 2016; date of current version August 7, 2017. This work was supported by the National Natural Science Foundation of China under Grant 61374192 and Grant 51575048 and the Beijing Natural Science Foundation under Grant 4162055.

L. Bi, Y. Lu, and J. Lian are with the School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081 China. (e-mail: bhxblz@bit.edu.cn; luyun_bit@126.com; lianjinling@bit.edu.cn).

X. A. Fan is with the Beijing Institute of Mechanical Equipment, Beijing 100854 China. (e-mail: anmengxiang@126.com)

Y. Liu is with the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI [illegible] USA (e-mail: yililiu@umich.edu).

Digital Object Identifier 10.1109/TNSRE.2016.2614003

This article has been accepted for publication in IEEE Transactions on Vehicular Technology. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVT.2023.3342151

Modeling Car-Following Behavior of Brain-Control Driving with Queuing Network Architecture

Yun Lu, *Member, IEEE,* Rong Su, *Senior Member, IEEE,* and Yili Liu

***Abstract*—Directly using brain signals to drive vehicles is called brain-control driving, which has the potential to help people with disabilities to acquire driving ability. Developing driver models of the brain-control driving can help understand and simulate the driver behaviors, which contributes to the development of the brain-control driving. In this paper, to simulate the car-following behavior of the brain-control driving, we propose two brain-control car-following models, i.e., CF-BCB and CF-BCI models. The two models are built by integrating a car-following decision model with a brain-control behavior model and a brain-computer interface (BCI) performance model in the queuing network (QN) cognitive architecture, respectively. The manual- and brain-control experiments are conducted to collect the car-following data of the ideal and real brain-control driving, respectively. Simulations with the proposed models are performed to mimic the ideal and real brain-control driving of different subjects. We compare the performance of the car-following decision model and two decision models extended from the intelligent driver model (IDM) and the full velocity difference model (FVDM), respectively. The comparison results show that the car-following decision model performs better than the two decision models in simulating the ideal brain-control driving. Besides, we prove the effectiveness of the proposed models in simulating the car-following behaviors of the brain-control drivers by comparing the simulation and experimental results.**

***Index Terms*—Brain-control driving, car-following behavior, queuing network, driver model.**

I. Introduction

DIRECTLY using brain signals to drive vehicles is called brain-control driving. Unlike the manual-control driving, the brain-control driving does not rely on muscles or peripheral nerves of human drivers. Such driving method, for one thing, has the potential to help people with disabilities to acquire driving ability. For another thing, it can free the limbs of healthy people by offering a new driving approach. The brain-control driving is performed by using brain-computer interfaces (BCIs), which can directly interpret the brain signals of humans into control commands [1]. Electroencephalography (EEG) has become the most popular technique to develop brain-computer interface (BCI) systems due to its low cost, high temporal resolution, and ease of use [2].

Several researches have explored to employ EEG-based BCIs to perform the brain-control driving. Göhring *et al.* [3] provided subjects with a BCI product from Emotive to drive an autonomous vehicle. Users could perform the speed and steering control when the vehicle was in a safe zone. In [4], a motor imagery-based BCI was designed for drivers to control the steering, movement, and engine of a vehicle at the velocity of about 5 km/h. Bi *et al.* [5] have investigated how to use a steady state visually evoked potential (SSVEP) BCI to control the steering of a vehicle with the speed of 2–3 m/s. In [6], we have developed a longitudinal driving system equipped with an SSVEP BCI for drivers to conduct the longitudinal control of a vehicle with the speed being no higher than 10 m/s. Lu and Bi [7] have explored the feasibility of applying an SSVEP-based BCI to perform the combined longitudinal and lateral control of a vehicle.

Owing to the performance limitations (e.g., accuracy limitation) of BCIs, the driving performance of the vehicles relying on BCIs is poor. A few studies have explored to build assistive control methods to improve the driving performance. Our previous works [8] and [9] have designed two shared control methods for the brain-controlled vehicles built in [5] and [6] to improve their lateral and longitudinal driving performance, respectively. Shi *et al.* [10] have built a control strategy by using the model predictive control (MPC) and fuzzy logic to improve the lateral performance of the brain-controlled vehicles. However, even with the assistance of the shared control techniques, the performance of the brain-control driving is not good enough. To further improve its performance, it is significant for vehicle designers and control engineers to understand and simulate the behavior of the brain-control drivers.

Developing driver models can help understand, compute, predict, and simulate the related driver behaviors [11], [12]. In practice, driver models can help increase the development efficiency of vehicle systems by decreasing the requirement to perform driver-in-the-loop experiments [13], [14]. They also contribute to the development of advanced driver assistance systems [...] Numerous researches have been carries out

Manuscript received January 10, 2023; revised May 20, 2023, August 30, 2023; accepted December 8, 2023. This work was supported by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-GC-2023-007).

(Corresponding author: Yun Lu)

Y. Lu and R. Su are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798. (e-mail: yun.lu@ntu.edu.sg; rsu@ntu.edu.sg).

Y. Liu is with the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI [...] USA. (e-mail: yililiu@umich.edu).

QN-HMN

task, the failed component will return to normal operation. This human-computer-machine system can be modeled as the simple queueing network model shown in Figure 6. For the 3-node (machine, computer, human) closed-series network, the state of the queueing system at any time instant t is a vector p(n1, n2, n3), representing the number of customers at node ni (i=1, 2, 3) at time t. For the system described above, p(n1, n2, n3) means that there are n1 machine components in normal operation, n2 being serviced by the computer, and n3 by the human. The total number of customers (denoted by S) should be a known quantity (e.g., S=k could mean that there is a total of k engines).

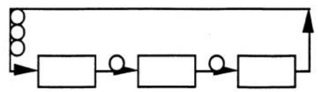


Fig.6. A closed queueing network model of human-computer interaction in a failure management system described in the text

p(n1, n2, n3) can be computed easily with the following set of equations, derived from the results of Jackson (1963) [34]:

$p(n1, n2, n3|S=k) = \omega^*(n1, n2, n3)/T^*(S=k);$

$\omega^*(n1, n2, n3) = \prod^{3}_{i=1}\prod^{k_i}_{j=1}(1/\mu ij);$

$T^*(S=k) = \sum\omega^*(n1, n2, n3)$, summed over (n1, n2, n3) with S=k;

where μij is the mean service rate of node i when there are j customers at node i. Apparently, the "service rate" of the machine (node 1) is the rate at which it causes machine components to fail. The values for uij are usually obtainable from measurements, specifications or historical data.

The above set of equations allow us to predict a number of interesting performance features of the system. For example, it is easy to compute the proportion of time during which the human operator will have at least one machine component to repair ($\sum$p(n1, n2, n3), summed over (n1, n2, n3) with n3>0 and S=k), or the proportion of time during which the machine will have at least two components working normally (e.g., at least two engines are running) ($\sum$p(n1, n2, n3), summed over (n1, n2, n3) with n1>1 and S=k).

We have extended the work to modeling more complicated systems involving more than one humans and more than one computers--a human-computer network. A specific example is a failure management system in which there are two type of machine component failures, each type is handled by a computer and then by a human operator. A possible scenario is that two computers and two human operators work cooperatively in a manner illustrated in Figure 7, where the "copilot" completes his/her task alone with a probability of p, but need to forward the problem to the "pilot" with a probability of (1-p), before the component is returned for normal operation.

In order to compute the queue length distributions, we need the routing probability of the customers--pij, the probability that a machine component will immediately visit node j after departing from node i, which is specified by the task structure. In Figure 7, we have,

p12 = q (the probability that a failure is of type 1),

p13 = 1-q (the probability that a failure is of type 2),

p54 = p (the probability that human operator 2 needs help from human operator 1),

p56 =1 - p (the probability that human operator 2 can complete his/her job alone)

p24 = p35 = p46 = p60 = p01 = 1

pij = 0, for all other i and j's.

The expected value of the number of appearances of node i on a routing is computed with the following recursive equation,

$e_i = p0i + \sum^{i}_{m=1} (e_m\, p_{mi})$

With a total of k machine components in the queueing system, p(n1, n2, n3, n4, n5) can be computed easily with the following set of equations, derived from the results of Jackson (1963):

$p(n1, n2, n3, n4, n5|S=k) = \omega^*(n1, n2, n3)/T^*(S=k);$

$\omega^*(n1, n2, n3, n4, n5) = \prod^{5}_{i=1}\prod^{k_i}_{j=1}(e_i/\mu_{ij});$

$T^*(S=k) = \sum\omega^*(n1, n2, n3, n4, n5)$, summed over (n1, n2, n3, n4, n5) with S=k;

A number of question can be answered with the computed queue length distributional values. The type of questions include the relative workload of operator 1 versus operator 2, the proportion of time during which the machine has at least c components operating normally, and the effects of changing network configuration or service rates.

Although the models are presented in the context of a network of human and computer agents interacting with each other toward a common goal, an area known as computer-supported cooperative work (CSCW). The same methodology can be applied to the broader area of human-computer networks, which also includes situations in which competitive or confrontive agents may compete with each other for

QN-HMN

limited network resources and cause delays in servicing other agents' processing needs.

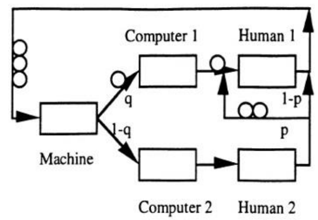


Fig. 7. A queueing network model of a human-computer network in the failure management system described in the text

Although a multitude of human-computer networking tools and CSCW applications have been developed, there is a substantial lack of predictive models and theories. As Schneiderman (1992) pointed out, this is a "vast uncharted territory: theories are sparse, measurement is informal, data analysis is overwhelming, and predictive models are nonexistent" ([35], p.391). The model presented in this section illustrates that queueing network methods could serve as a useful tool for establishing performance theories and predictive models of human-computer networks and for establishing theory-guided, systematic ways of performance measurement and analysis, particularly the issues of concern involve timing, scheduling and resource allocation.

The models presented in Figures 6 and 7 are currently being evaluated with lab experiments using a simulated failure management system and human subjects. We are also in the process of preparing experiments to validate a model of human-computer network with competing agents.

We hope that this article has illustrated the potential power of queueing network methods in establishing new models of human cognition, human performance and human-computer interaction on various analysis levels, and in establishing an integrated, computational framework for unifying some currently isolated models.

REFERENCES

[1] R. Disney and D. Konig, "Queueing networks: A survey of their random processes," SIAM Review, vol-27, 335-403, 1985.

[2] L. Kleinrock, "Queueing Systems," New York: Wiley, 1975.

[3] Y. Liu, "Visual scanning, memory scanning, and computational human performance modeling," Proc. of the Human Factors Society 37th Annual Meeting, 1993.

[4] Y. Liu, "A queueing network model of human multi-task performance," Tech. Rep. Univ. of Michigan, Dept. of IOE. 1993.

[5] J. Jackson, "Networks of waiting lines," Oper. Res., vol-5, pp.518-521, 1957.

[6] O. Boxma and H. Daduna, "Sojourn times in queueing networks," In H. Takagi (Ed.), Stochastic Analysis of Computer and Communications Systems, p.401-450, 1990, North Holland.

[7] J. Buzen, "Fundamental operational laws of computer system performance," Acta Informatica, vol-7, pp.167-182, 1976.

[8] P. Denning and J. Buzen, "The operational analysis of queueing network models," Computing Surveys, vol-10, pp.225-261, 1978.

[9] S. Sternberg, "The discovery of processing stages: Extensions of Donders's method," Acta Psychologica, 30, p.276-235, 1969.

[10] R. Pachella, "The interpretation of reaction time in information processing research," In B. Kantowitz (Ed.), Human information processing: Tutorials in Performance and Cognition, p.41-82, 1974, Hillsdale, N.J.: Erlbaum.

[11] W. McGill and J. Gibbon, "The general-gamma distribution and reaction times," J. of Math. Psych., 2, 1-18, 1965.

[12] J. McClelland, "On the time relations of mental processes: An examination of systems of processes in cascade," Psychological Review, 86, 287-330.

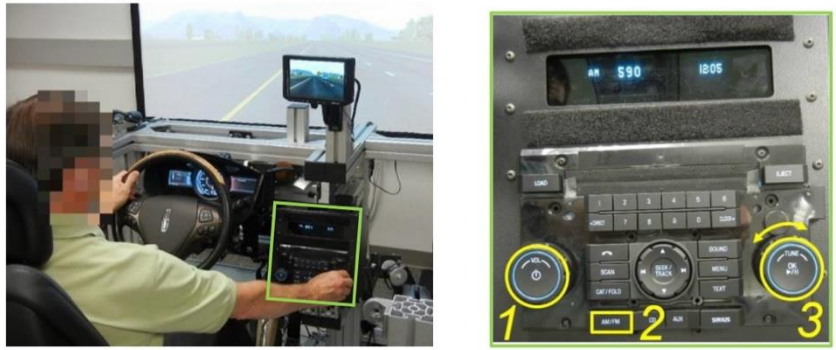
[13] J. Miller, "A queue-series model for reaction time, with discrete-stage and continuous-flow models as special cases," Psychological Review, 100, 702-715, 1993.

[14] J. Townsend and F. Ashby, "The Stochastic Modeling of Elementary Psychological Processes," Cambridge: Cambridge Univ. Press, 1983.

Queueing Network (QN) Models of Human Behavior (MHB) QN-MHB

1. RT: Reaction Time (QN-RT) and Mental Structure
2. RT and Accuracy (QN-RMD) (Mental Structure vs State of Mind)
3. Procedural Tasks (QN-MHP or QN-MHP-BE)
4. Complex Cognition Tasks (QN-ACTR)
5. Visual Attention tasks (QN-NSEEV) (QN-RLEM)
6. Manual or Continuous Control tasks (QN-Control: Classical/Modern)
7. Basic Body Motion tasks (QN-MTM)
8. Mind-Body System (QN-MBS)
9. Neural level (QN-Neural: Indexes and Neural Networks)
10. Nervous and Endocrine Systems (QN-NES)
11. Multi-Person Multi-Machine QN (QN-HMN)
12. Engineering Applications in various domains

Note: relations with **Task Network** Methods/Tools such as Micro Saint #, IMPRINT)



(a) Tuning radio using the knob (b) A zoom-in view of the physical panel

Figure 31. Task Procedure using the knob: (1) press the power button, (2) press the AM/FM button, (3) turn the knob to decrease or increase the frequency shown on the display ("590" as in the picture)

(a) Tuning radio on the touch screen (b) Click the "Entertainment" button

(c) Click the "FM" then "Direct Tune" button (d) Enter the radio frequency

Figure 32. Task procedure using the virtual buttons

Fig. 9. Touch screen UI used in the experiment (left) and its digital mockup (right).

Fig. 10. Physical panel used in the experiment (left) and its digital mockup (right).

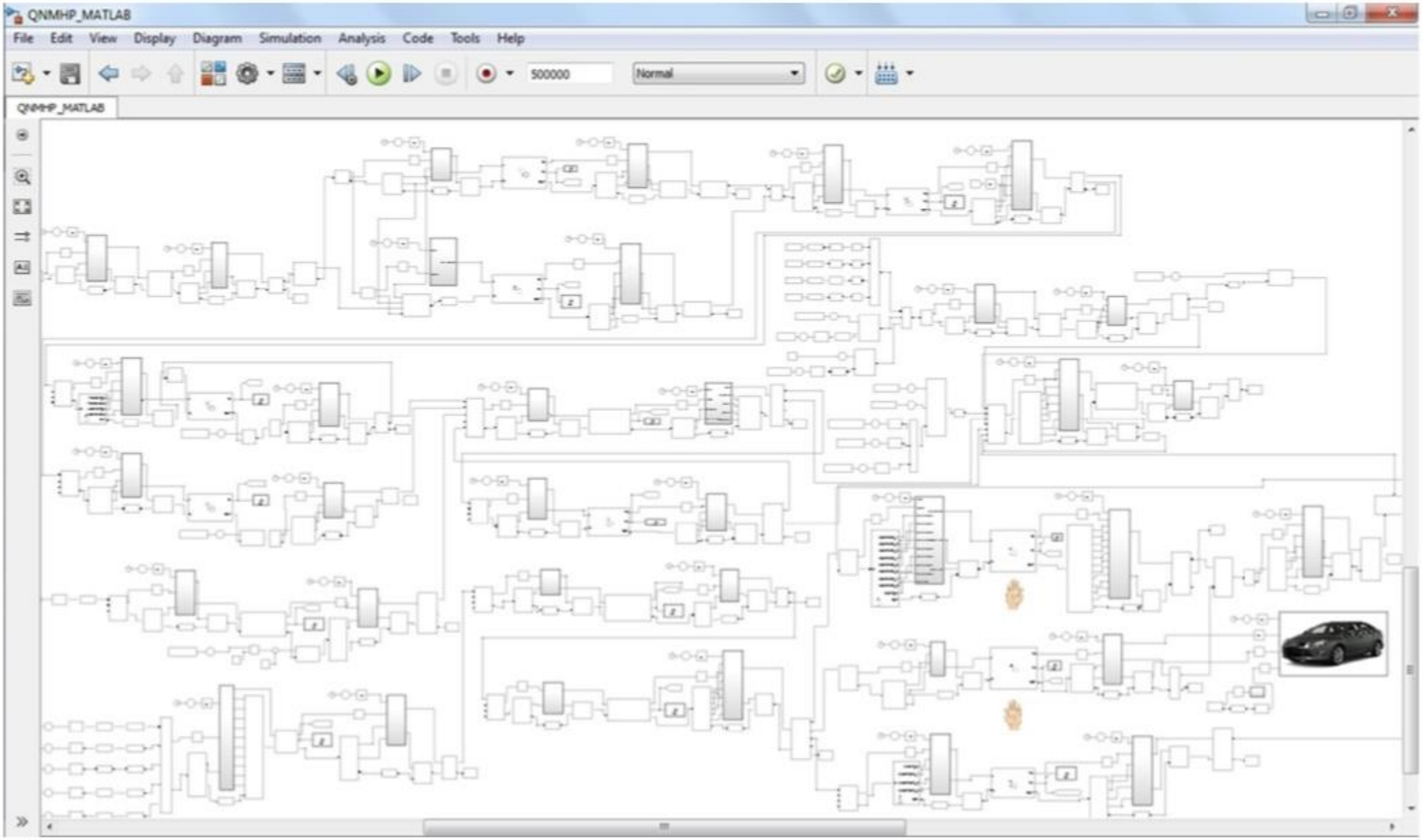
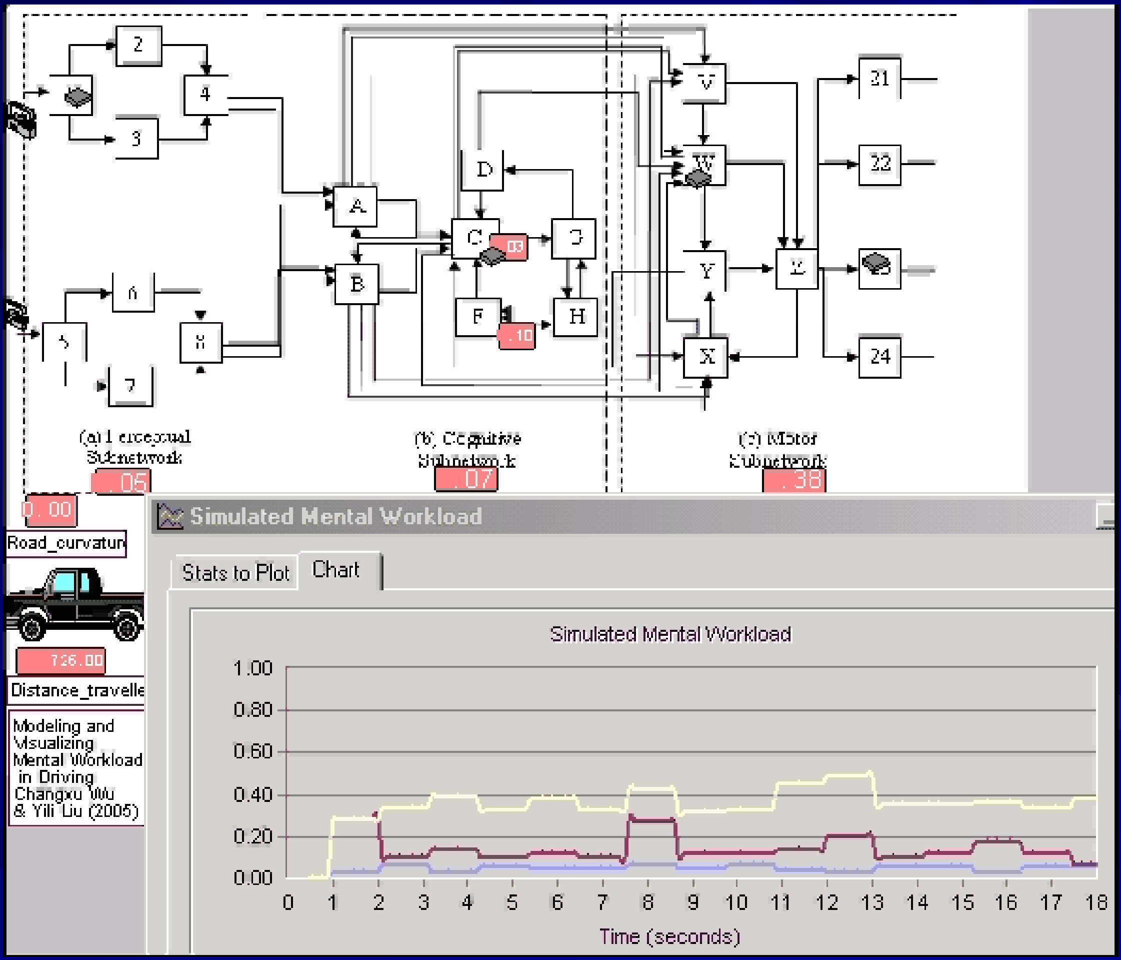


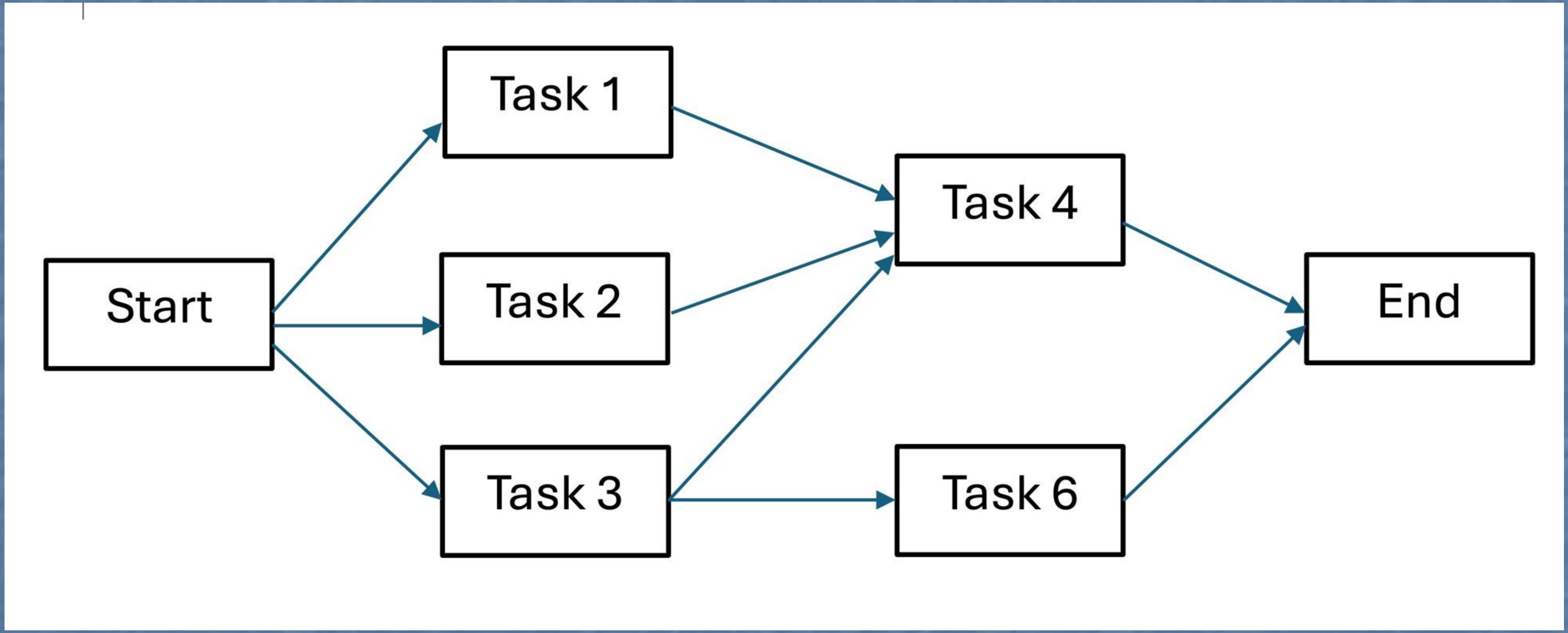
Fig. 2. A screenshot of the QN-MHP model implementation in MATLAB/Simulink.

Visualizing Mental Workload (perceptual, cognitive, motor loads)

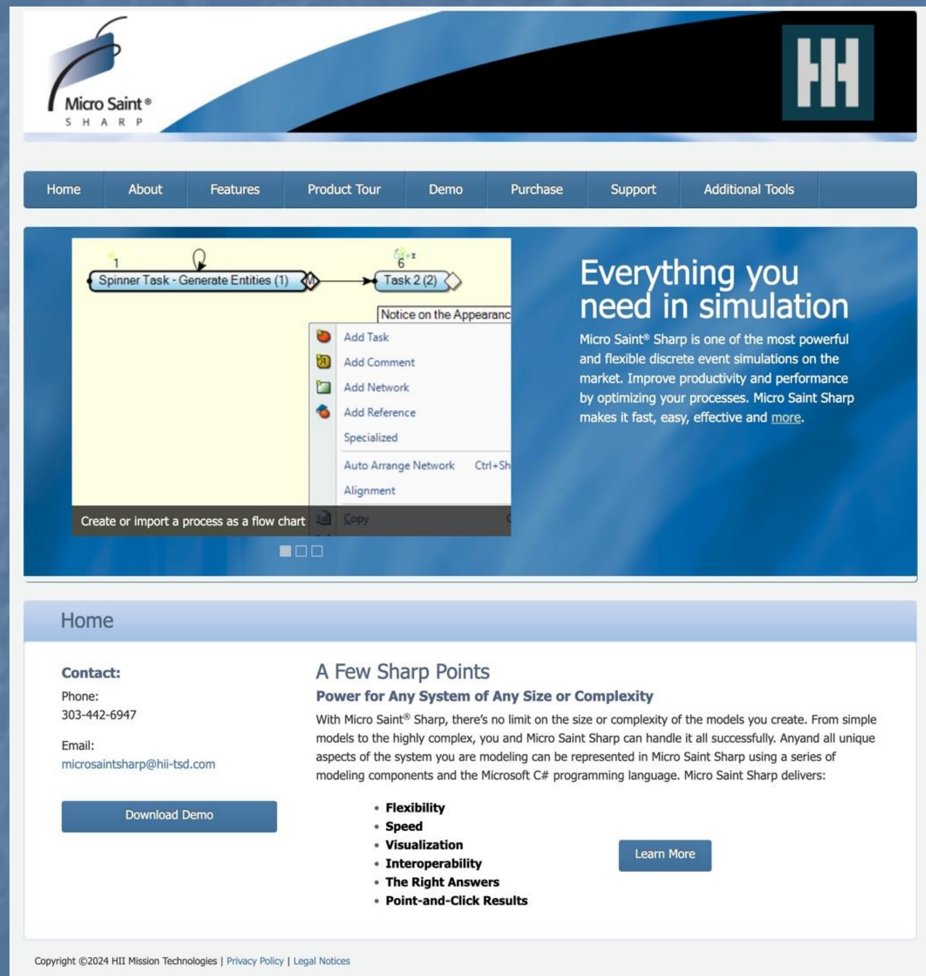


QN-MHP driving

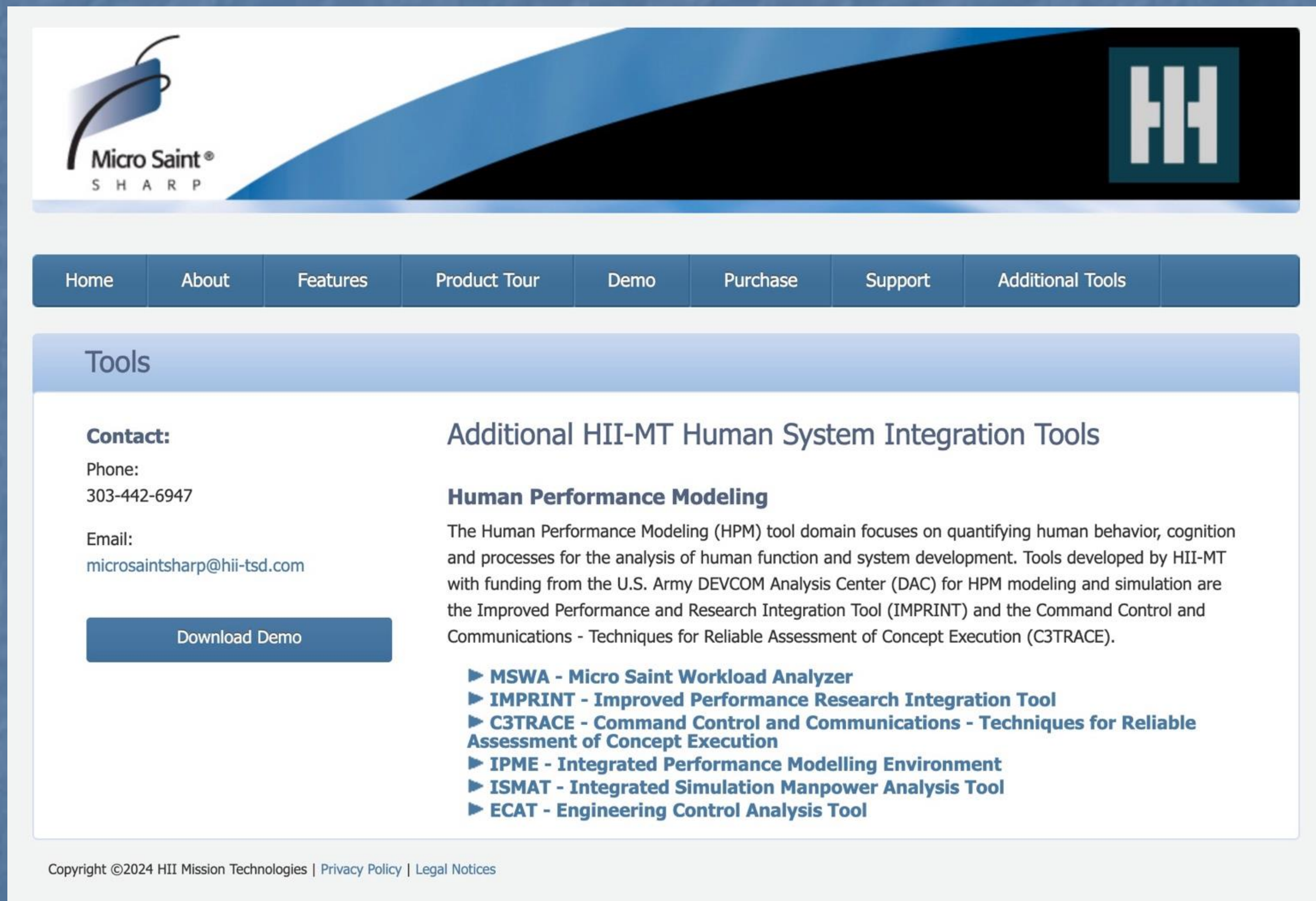
Task Network
(Project network, PERT, CPN, Mission Network, etc.)

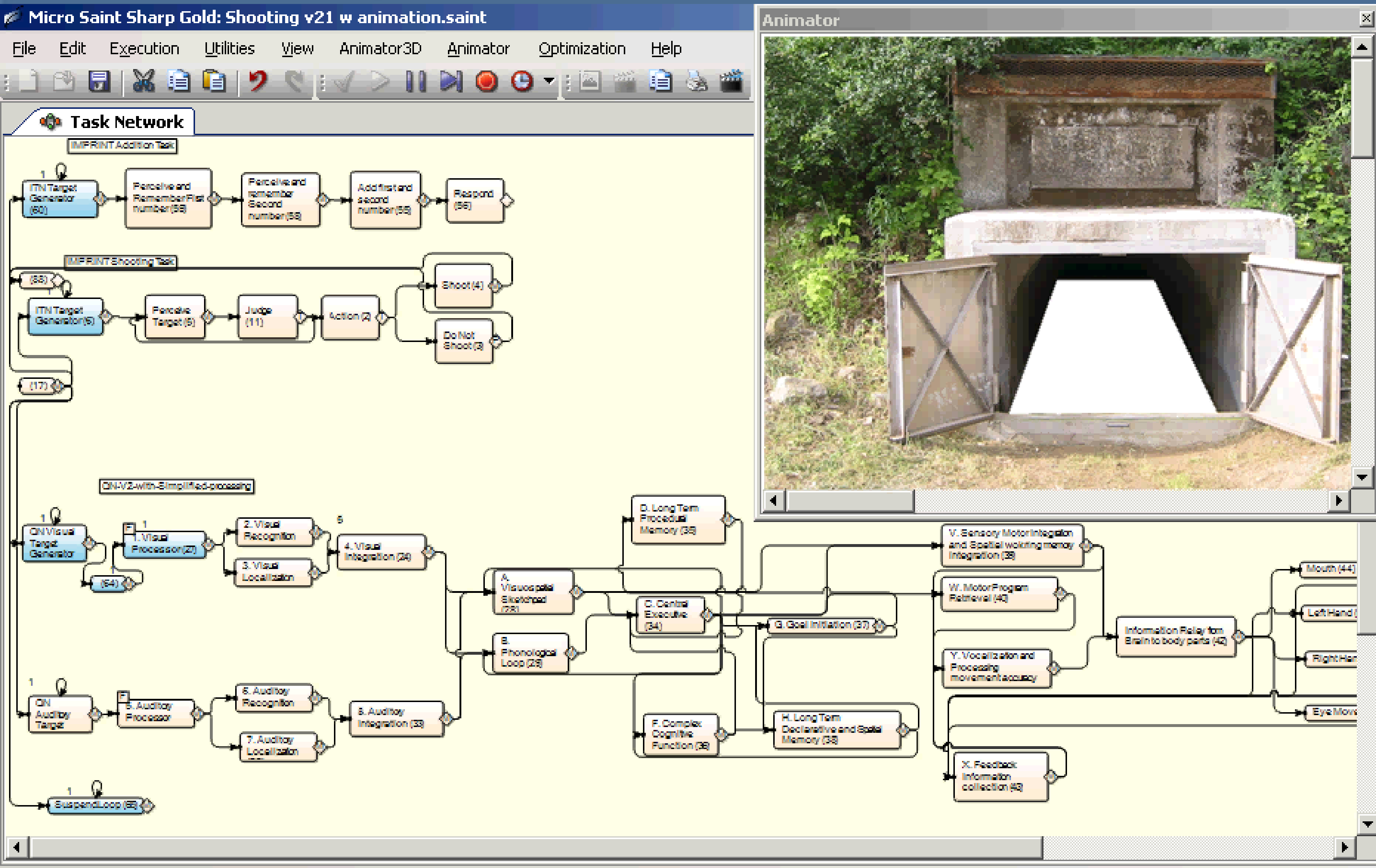


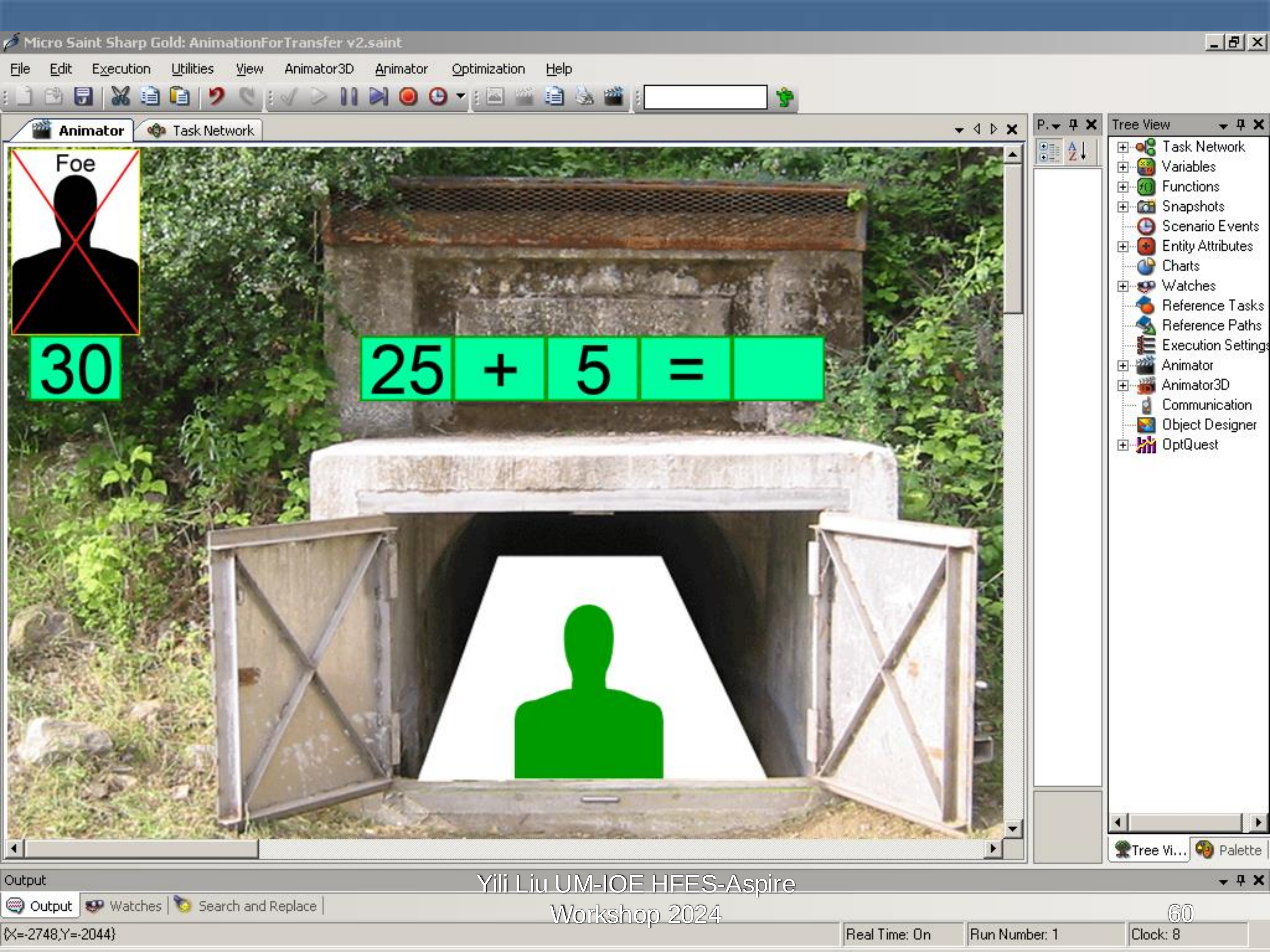
Relationship between QN and Task Network Modeling



Relationship between QN and Task Network Modeling




Micro Saint Sharp Gold: Shooting v21 w animation.saint
File Edit Execution Utilities View Animator3D Animator Optimization Help
Task Network
Animator
IMPRINT Addition Task
IMPRINT Shooting Task
QN-V2-with-Simplified-processing


Micro Saint Sharp Gold: AnimationForTransfer v2.saint
File
Edit
Execution
Utilities
View
Animator3D
Animator
Optimization
Help
Animator
Task Network
Foe
30
25
+
5
=
Tree View
Task Network
Variables
Functions
Snapshots
Scenario Events
Entity Attributes
Charts
Watches
Reference Tasks
Reference Paths
Execution Settings
Animator
Animator3D
Communication
Object Designer
OptQuest
Tree Vi...
Palette
Output
Output
Watches
Search and Replace
{X=-2748,Y=-2044}
Real Time: On
Run Number: 1
Clock: 8

Queueing Network Modeling of Human Performance

in Complex Cognitive Multi-task Scenarios

by

Shi Cao

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in the University of Michigan
2013

Modeling Dual-Task Concurrency and Effort in QN-ACTR and IMPRINT

by

Christopher Jason Best

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2013

Doctoral Committee:

Professor Yili Liu, Chair
Assistant Professor Victoria Booth
John F. Locket III, US Army Research Laboratory
Professor Nadine B. Sarter

An Integrated Cognitive Architecture for Cognitive Engineering Applications

Shi Cao and Yili Liu
University of Michigan
Ann Arbor, Michigan 48109 USA

The increasing complexity of computational cognitive architectures may increase both their modeling capabilities and their difficulty to learn and use as cognitive engineering tools. This paper reports our work dedicated to enhance the usability and the cognitive engineering applicability of a complex computational cognitive architecture called QN-ACTR, which integrates two complementary architectures Queueing Network and Adaptive Control of Thought-Rational. The aim is to provide an easy-to-use interface and intuitive modeling that support both inexperienced and experienced users in using this complex and powerful architecture. The process of model development is greatly simplified with improved visualization and validation methods. The results were examined using heuristic evaluation. The benefits and practice implications are discussed.

INTRODUCTION

Cognitive models can be used to support cognitive engineering. Compared with other forms of cognitive models such as verbal frameworks and pure mathematical models, cognitive architectures are particularly useful for complex cognitive engineering applications, because they unify a wide range of cognitive theories (Newell, 1990) and can computationally simulate human-machine interactions (Byrne & Pew, 2009; Schunn & Gray, 2002). For example, Adaptive Control of Thought-Rational (ACT-R, Anderson et al., 2004), a cognitive architecture that has incorporated many of the theoretical advances of cognitive science over the past decades, has been applied to cognitive engineering analyses of human-machine interactions including airport runway navigation, driving performance, and human-computer interactions (for a review, see Gray, 2008).

In the recent years, cognitive architectures are becoming increasingly integrated and complex in terms of having more components and interactions between components, requiring the use of knowledge description languages, and involving a large number of parameters. This complexity may increase both modeling capabilities and the difficulty to learn and use them as cognitive engineering tools. For example, building useful models in ACT-R requires a considerable amount of training and practice. The basic theory and syntaxes of ACT-R can be learned by reading a seven-unit tutorial and practicing with examples, which are often covered in a seven-day short course. The model description of displays and controls is written in the Lisp language, and therefore a modeler must also gain reasonable Lisp programming skills. Adjusting model parameters could also be very difficult, because the effect of changing a parameter may be buried deep in the text-based output traces. Currently, most users of cognitive architectures are expert researchers of cognitive modeling. The usage among cognitive engineers in the industry is very limited.

To emphasize the need for the usability development of cognitive architectures for cognitive engineering, Pew (2008) pointed out three challenges for researchers in this field, including the needs for (1) simplified model development, (2) better capabilities for articulating and visualizing how the models work, and (3) model validation.

Recently, several efforts have been made to address these challenges. G2A (Amant, Freed, & Ritter, 2005) and ACT-Simple (Salvucci & Lee, 2003) were developed to automatically translate GOMS (Goals, Operators, Methods, and Selection rules) style operators into ACT-R production rules. Incorporating ACT-Simple, CogTool (John, Prevas, Salvucci, & Koedinger, 2004) simplified the construction of human-computer interaction tasks, allowing the modeling of web browsing tasks by user demonstration with the mouse and the keyboard. Integrating ACT-Simple and an ACT-R driving model (Salvucci, 2006), Distract-R (Salvucci, 2009) simplified the construction of models for human interaction with in-vehicle devices in driving scenarios. Using Visual Basic Application in Excel, a click-and-select user interface has been developed in Queueing Network-Model Human Processor (QN-MHP, Wu & Liu, 2008). It allows users to build QN-MHP models without learning any simulation language. Usability tests showed that this click-and-select interface can save time and reduce errors in model development (Wu & Liu, 2009). In addition, easy-to-use user interfaces have also been developed in E-GOMS (Gil, 2010) and SANLab-CM (Patton & Gray, 2010).

The previous efforts have simplified model development, reducing or eliminating the need for learning a modeling language. However, each of them still has limitation in one or more of the following aspects.

- The simplification of modeling work comes at the cost of limiting the task displays and controls that can be model to a limited set of tasks.
- The flexibility to construct customized models with customized parameters is limited, for example, not being able to define the road curvature of each road segment as in a human driving experiment.
- Human information processing that can be modeled is limited to procedural and perceptual-motor processes, lacking the capabilities to model complex cognitive tasks such as learning, decision making, and sentence comprehension.

writing texts in tables and selecting options from menus. Users start from selecting the single or dual task scenario or loading demonstrations (Figure 3). Previous research models are included as demos (samples), such as ACT-R Tutorial models, driving models, and a transcription typing and reading dual-task model. New models can be made by modifying existing demos.

When a single task is selected with a template such as the Day-Block-Trial template for discrete-event experiments or the World3D template for driving, a task setup window will appear, asking users for the information needed in the experiment setup. When a dual-task is selected, two windows will appear each defining a task. For example, the Day-Block-Trial template in MSA asks for configuration settings such as whether a display stage in a trial will be terminated when all responses are detected (Figure 4) and setup details such as the number of trials in a block and the type of a display item (e.g., text, line, button, and tone) (Figure 5). The World3D template defining a driving task asks for road and car details such as lane width, road type, road length, and other cars' speed (Figure 6).

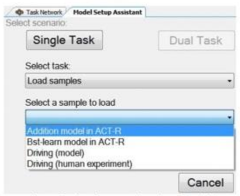


Figure 3. Screenshot of selecting a task using Model Setup Assistant.

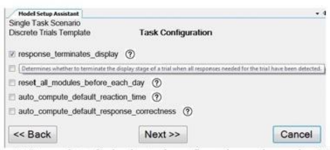


Figure 4. Screenshot of selecting task configuration options using Model Setup Assistant.

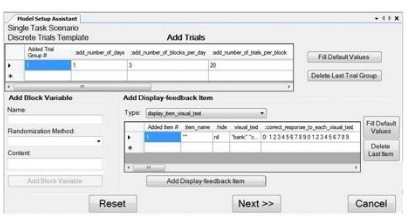


Figure 5. Screenshot of defining a discrete task using Model Setup Assistant.

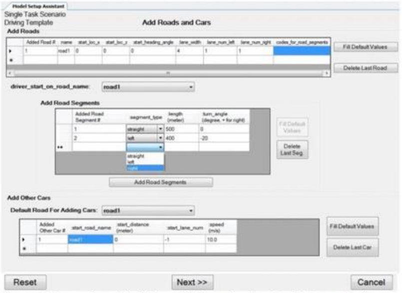


Figure 6. Screenshot of defining a driving task using Model Setup Assistant.

After defining the task, MSA can also assist users to define the mind and the parameter parts of a model. Since these syntaxes in QN-ACTR are the same as in ACT-R, if existing ACT-R codes are available, users can simply copy the ACT-R codes and paste into a QN-ACTR model. If no existing code available, users can define the mind, including chunks (Figure 7a) and production rules (Figure 7b), and set parameters (Figure 7c) with the assistance of MSA, by filling in tables and selecting from lists without the need to learn the knowledge description language used in ACT-R.

The model generated by MSA is also written in syntaxes. The resulted syntaxes can be saved, edited, or directly used to run the model. Simple modification of a model such as changing a few parameters can be easily achieved by directly editing the syntax file.

Visualization of 3D dynamic tasks

Previous work has developed the visualization of mental information processing, discrete experiment displays and controls, and the multi-dimensional mental workload (Cao & Liu, 2011a, 2011b). A new feature added to the visualization capabilities of QN-ACTR in this study is visualizing 3D dynamic tasks.

Using Animator3D in Micro Saint® Sharp, 3D dynamic tasks such as driving in single or dual task scenarios can be visualized in real time while the model is performing the task, which allows intuitive observation of model performance. The system refresh rate can be set by the user (10 ms by default). System dynamics such as speed, steering angle, and lateral deviation are visualized and recorded. Figure 8 illustrates that the model is driving a car while performing an arithmetic addition task. The model is following the car in the right lane and is visually focusing on the car. At the same time, the model is speaking "three" in response to the question of "1 + 2," which is displayed through the auditory channel and visualized on the right hand side of the figure.

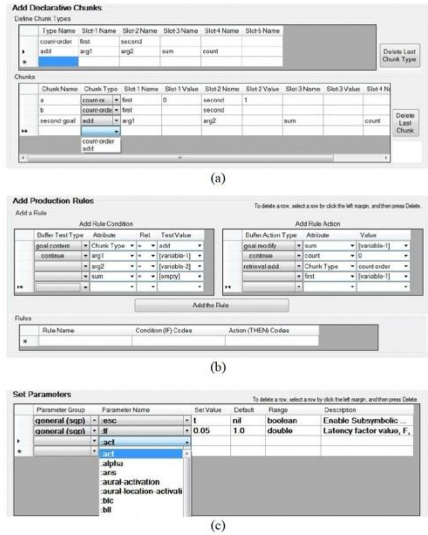


Figure 7. Screenshot of defining the mind and the parameter parts of a model using Model Setup Assistant, including (a) chunks, (b) production rules, and (c) parameters.

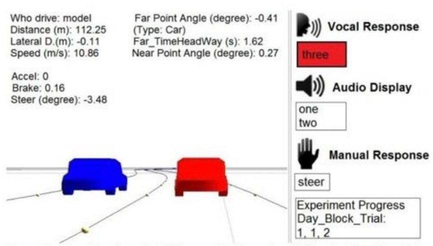


Figure 8. Visualization of a driving and arithmetic addition dual-task in QN-ACTR.

Integrated human experiment interface

The same task interface with which the model interacts can also serve as the interface for human participants to complete the same tasks. We have developed a human driving interface in QN-ACTR that supports simulated driving experiments with steering wheels and pedals. This feature allows the model and the human to perform and be compared in the same tasks with identical interfaces, with no need to replicate the real world experiment system in the modeling environment. Using the same experiment platform avoids any discrepancy between human and model tests due to the experiment setup.

FINDINGS

The usability development of QN-ACTR is evaluated using Nielsen's ten heuristics for user interface design (1994).

Visibility of system status. MSA always shows the stage of model development at the top-left corner. The visualization of the mind and the task keeps users informed about what is going on in the model during the simulation. Buttons in MSA are dimmed and disabled when their actions cannot be performed in some cases. Program responses and feedbacks are immediate with no delay.

Match between system and the real world. All the column headers in MSA tables and the items in menus use self-explanatory phrases without abbreviation. The steps of modeling in MSA follow the logical order shown in Figure 2. Full names and detailed descriptions are shown for each abbreviated ACT-R parameter name (Figure 7c).

User control and freedom. MSA supports undo (e.g., change the road name, delete a chunk, and reset a table) and redo (e.g., go back to the previous stage, and then go next again). A cancel button is provided at each stage to exit the setup at any time, and then users can restart MSA if needed.

Consistency and standards. Definitions and names are used consistently throughout all modeling steps. Tables and menus follow similar layouts and styles. Button position is the same between templates and stages.

Error prevention. The use of menu selection in MSA tables prevents the input of invalid items. Table cells automatically perform validation check, and users are notified when an input is of an invalid type or out of the valid range. Duplicated names assigned by users (e.g., chunk names) are automatically revised to prevent run-time errors. Syntax errors are also reported before the simulation starts.

Recognition rather than recall. MSA provides menus for users to select their options and tables to fill in. Model developing knowledge is provided to users in the interface. For example, users do not have to learn any modeling syntax. Instead, they can describe the model in natural language and fill in blanks or select items (Figure 7a, b). The default value, valid range, and description of model parameters are displayed for the users (Figure 7c).

Flexibility and efficiency of use. The syntax method and MSA cater to both inexperienced and experienced users. Experienced user can speed up the modeling work by directly copying and editing syntaxes. Syntaxes for the mind and the parameters can also be directly copied from ACT-R codes.

Aesthetic and minimalist design. MSA tables and menus are organized and aligned in groups. Introductions and explanations are concise.

Help users recognize, diagnose, and recover from errors. Error messages are expressed in plain language (no codes) and precisely indicate the problem. For example, "Error! Set General Parameters needs para_name: :lf to be a double rather than: nil."

Harper, 2007). This scenario used the same DISALT shooting testbed and provided behavioural data to model the shooting task. Production rules were defined to model the shooting-only task as the process of visual searching, manual aiming, and pulling the trigger.

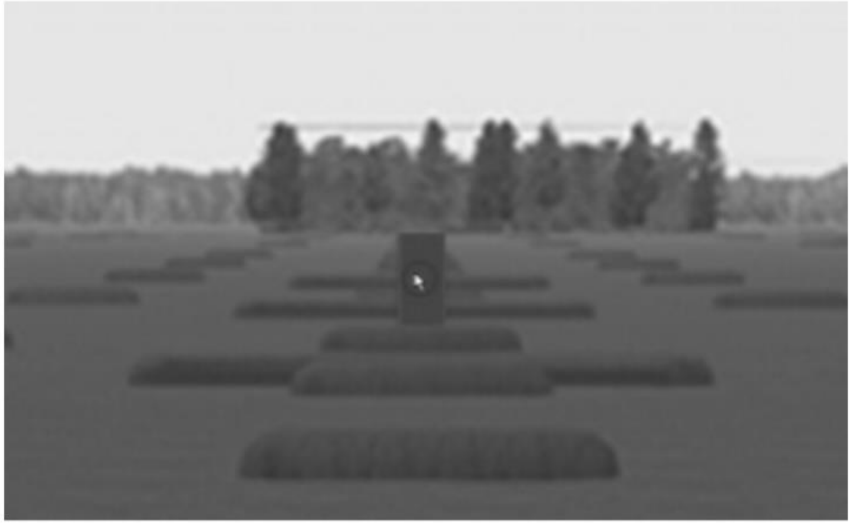


Figure 4 Visualization of the task environment with an enemy-target in QN-ACTR. The cursor represents the iron sight aiming. Background picture from (Scribner et al., 2007).

Relationship between QN and Task Network Modeling

CHAPTER

30

Queuing and Network Models

Yili Liu

Abstract

Comprehensive and computational models of human performance have both scientific and practical importance to human-machine system design and human-centered computing. This chapter describes human performance models that are based on queuing and network theories. Queuing-based models and network-based models were initially developed as two separate schools of models, as summarized in the first part of the chapter. Recent work based on queuing networks not only integrates the two schools of models into a unified framework but also allows integration of several other schools of approaches such as symbolic models, as described in the second part of this chapter.

Key Words: queuing models, network models, queuing network models, cognitive modeling, human-machine interaction models, systems modeling

Introduction

The increasing complexity of advanced human-machine systems makes it necessary for system designers to consider human capabilities and limitations as early as possible in system design. In order to reduce risks associated with poor task design with appropriate tools and methods for task analysis and function allocation, it is important to develop models of human performance and human-system interaction that are comprehensive, computational, science-driven, and application-relevant.

Models of human performance and human-system interaction should be comprehensive to capture the whole range of concurrent perceptual, cognitive, motor, and communication activities of human-system performance (also see Byrne, this handbook). These models should be computational and computerized to allow quantitative and rigorous simulation and analysis of design alternatives and scenarios. These models should be science-driven, with deep roots in and strong connections with cognitive science theories and principles. These models should also be application-relevant, striving to tackle and solve practical design problems, with an engineering philosophy that having an "imperfect" or approximate solution is better than no solution at all.

Human performance models for complex human-machine systems must also take into account the fact that operators in human-machine systems often need to perform a number of concurrent activities at once (see also Salvucci, this handbook). Examples of multitask situations abound and include an automobile driver who has to ensure the smooth operation of a vehicle while time-sharing between the instrument panel and the forward view of the roadway, and a traffic controller who has to divide attention between various visual and auditory sources of information while making time-critical decisions and performing intensive communications activities.

Many computational models have been proposed to model multitask performance and address the nature and the cause of task interference, in

449

Relationship between QN and Task Network Modeling

addition to various conceptual theories and qualitative models. This chapter focuses on computational models of multitask performance that are based on queuing and network theories, which represent some of the most prominent approaches in computational multitask modeling.

The specific contribution of this chapter to the challenges of engineered or technological systems is its emphasis and demonstration of the importance and value of queuing and network models in human-machine system design. The chapter first summarizes queuing-based models and network-based models, which were initially developed as two separate schools of models, and then describes recent work based on queuing networks that not only integrates the two schools of models into a unified framework but also allows integration of several other schools of approaches such as symbolic models.

Single-Server Queuing Models

Historically, computational modeling of multitask performance started with the school of computational modeling that we call single-server queuing models. This school of models encapsulates computationally a prominent conceptual theory of multitask performance called the single-channel theory of selective attention. Its roots can be traced to the single-channel theory of human information processing originally proposed by Craik (1947), which assumes that the human information processing system has bottlenecks that can process only one stimulus or piece of information at a time, and thus the system functions through a series of selections about which stimulus or piece of information to process (see also Broadbent, 1958; Deutsch & Deutsch, 1963; Welford, 1967).

The single-channel psychological theory of selective attention has been the fundamental basis of numerous engineering models of human performance (Carbonell, 1966; Rouse, 1980; Senders, 1964; Senders & Posner, 1976; Schmidt, 1978). These engineering models postulate that the human is a single-channel processor or a time-shared computer with a single central processing unit (CPU), which quickly switches and allocates its processing capacity among a variety of tasks in a sequential and all-or-none fashion. The models view human multitask performance as a single-server queuing problem or multitask sequencing problem in which multiple tasks or diverse sources of information are queued for service from the single-server human information processing system. For a comprehensive and detailed review of this school of models, see Liu (1997).

Task Network Models

Another school of engineering models of human performance that has had a long history and wide range of successful applications is the task network models. Starting with the systems analysis of integrated networks of tasks (SAINT) modeling methodology developed by Siegal and Wolf (1969), the task network approach models the human interaction with the environment as a sequence of tasks (also called paths) and acknowledges the existence of alternative paths to accomplish a goal or different goals in certain circumstances. These alternative paths form a task network. Parallel paths in a task network represent alternatives rather than concurrence of processing.

The modeling methodology of SAINT has been substantially and significantly extended into a family of network-based models, prominent among them includes Micro Saint Sharp, which is a general-purpose, discrete event simulation software tool that has been used successfully in many areas including human factors and the military, manufacturing, and service sectors (Laughery, 1989). Micro Saint was also used as the platform to develop the flagship task network modeling tool of the Army Research Lab (ARL) called IMPRINT (Improved Performance Research Integration Tool). IMPRINT is arguably the most powerful of the Army's Human System Integration (HIS) tools developed over the past two decades. It is a Windows-based, dynamic, stochastic, discrete event-modeling framework. When certain assumptions hold—that is, when the system of interest can be adequately described by task activities and networked sequencing, when dynamic processes and random variability are of interest, and when any continuous tasks can be fairly transformed into discrete tasks—then IMPRINT is an appropriate tool to use to represent and analyze soldier-system performance. As a system design and acquisition support tool, IMPRINT can be used to help set realistic system requirements, to identify soldier-driven constraints on system design, and to evaluate the capability of available manpower and personnel to effectively operate and maintain a system under environmental stressors. IMPRINT is also used to target human performance concerns in system acquisition, to estimate user-centered requirements early, and to make those estimates count in the decision-making process (Hawley, Lockett, & Allender, 2005).

Relationship between QN and Task Network Modeling

In spite of their differences, one of the common features of these models is their reliance on the fundamental assumption that humans can process only one piece of information at a time. Human multitask performance is modeled as a process of selecting tasks for sequential action according to some service discipline or cost function, which is usually based on the assumption that there is a mental cost to switching attention and/or there is a cost of being unable to attend to a critical instrument in a timely fashion (Rouse, 1980; Sheridan, 1972). Another common characteristic of these models is their focus on time as the underlying dimension and the metric of processing. Time is what is competed for by multiple tasks in a serial fashion, and completion time defines the difficulty or demand of each task or task component. The models are relatively silent as to the intensity aspects of task demand.

Recent work based on queuing networks not only integrates the two schools of models into a unified framework but also allows integration of several other schools of approaches, such as symbolic models (e.g., Anderson et al., 2004) and multiple resources theories (Wickens, 1984; Wickens & Liu, 1988), as described in the second part of this chapter.

Queuing Network Models

As described above, in order to support human-machine system analysis and design, it is important to develop models of human performance and human-system interaction that are comprehensive, computational, science-driven, and application-relevant. Along this line of research, a queuing network (QN)-based unified theory and computational architecture of human performance and human-system interaction has been developed that simultaneously meets the criteria listed above: comprehensive, computational, science-driven, and application-relevant. As reflected in the name "queuing networks," this approach explicitly considers both the queuing and the network aspects of human performance and modeling. Several major steps have been taken along this direction, each producing significant results and generating unique insights on human performance modeling in general and the role of queuing networks in human performance modeling in particular.

The following sections of this chapter first summarize accomplishments along this line of research and then discuss the next steps of research. The queuing network approach to human performance modeling was developed in several steps, starting from basic psychological functions and fundamental psychological issues and then moving to more complex tasks in human-machine systems. More specifically, since reaction time (RT) is arguably regarded as the most important and most widely used human performance measurement, the first step of the QN work was the successful development of a QN theory of reaction time (RT) that integrates the influential psychological architectural RT models as special cases, including the serial discrete-stages, the serial continuous-flow, and the discrete network models (such as the critical path network model). Further, the QN models cover a broader range of mental architectures and can be subjected to well-defined empirical tests. In the second step, the focus was on the relationship between RT and response accuracy, whose tradeoff is one of the fundamental characteristics of human performance. In this step of QN work, the architectural RT models and the sequential information sampling RT/accuracy models are unified through QN-RMD (Reflected Multidimensional Diffusions). Specifically, the "state" of a K-server QN of mental architecture is represented as a reflected diffusion space of K dimensions, in which "reflecting barriers" reveal architectural constraints, while "absorbing barriers" represent accuracy-related response criteria. QN-RMD moves beyond the current one-dimensional random walk/diffusion/accumulator models that have successfully accounted for but are limited to single-stage fast responses.

In the third step, QN-MHP (Model Human Processor) was developed to bridge the mathematical and the symbolic models of mental architecture and to support mathematical modeling and real-time generation of multitask performance and mental workload. QN-MHP expands the three discrete serial stages of perceptual, cognitive, and motor processing in MHP into three continuous-transmission subnetworks of servers, each performing distinct psychological functions specified with a procedural/symbolic language. Multitask performance and workload emerges as the network behavior of multiple streams of information flowing through a network. QN-MHP has been applied to generate and model a variety of tasks including the psychological refractory period, visual search, transcription typing, and driving a vehicle simulator.

QN Architecture of RT: Integrating Architectural and Information Transmission Models

Historically, the first groups of computational models that examine human performance and the

Relationship between QN and Task Network Modeling

IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 27, NO. 2, MARCH 1997 195

Queueing Network Modeling of Human Performance of Concurrent Spatial and Verbal Tasks

Yili Liu, *Member, IEEE*

***Abstract*— This article describes a three-node queueing network model of human multitask performance to account for interferences between concurrent spatial and verbal tasks. The model integrates considerations of single-channel queuing theoretic models of selective attention and parallel processing, multiple-resource models of divided attention, and provides a computational framework for modeling both the serial processing and the concurrent execution aspects of human multitask performance. The single-channel and the multiple-resource concepts and their applications in engineering models are reviewed. Experimental evidence in support of the queueing network model is summarized. The potential value of using queueing network methods to integrate currently isolated concepts of human multitask performance and in modeling human machine interaction in general is discussed.**

I. Introduction

One of the common characteristics of an operator's task in human–machine systems is the need to perform a number of concurrent activities at once. Examples of multitask situations abound and include an automobile driver who has to ensure the smooth operation of a vehicle while time-sharing between the instrument panel and the forward view of the roadway, and a traffic controller who has to divide attention between various visual and auditory sources of information, while making time-critical decisions and performing intensive communications activities. The requirements for processing multiple sources of information often push the operators in multitask human machine systems to the upper bound of their attention capabilities.

Fortunately, the increasing power and sophistication of hardware and software technology are providing additional options for human–machine system design that could take into account the characteristics of the human operator. In this regard, as indicated by a recent report of the Committee on Human Factors, National Research Council, comprehensive engineering-based predictive models of operator performance and workload in complex multitask systems become increasingly important [3]. These models can help assess the impact of the technological infusions and determine the most effective design before a system is configured, and will allow the human factors professionals to communicate their knowledge to the engineering community more effectively in a language that is compatible with the designers' existing terminology and conceptual base.

Many predictive models of multitask performance have been proposed to address the nature and the cause of task interference. Prominent among these models are the single-channel serial processing models and the multiple-resource parallel processing models. The two schools of models have fundamental differences in their views of the nature of multitask performance and in their research and modeling methodology. The single-channel serial processing models treat multitask performance as an issue of task selection and scheduling: human information processing systems can only attend to one task at a time, and multitask performance relies on the rapid switching of attention among the tasks competing for attention [9], [36], [45]. The multiple-resource parallel processing models, in contrast, treat multitask performance as an issue of parallel allocation and division of processing resources among simultaneous tasks: multiple tasks can be processed at the same time as long as the total demand does not exceed the limit of attentional capacity or processing resources [59].

Until recently, there has been a substantial gap between the two schools of models. As models of human behavior, both schools of models have received substantial support from a multitude of experimental studies. But at the same time, it has become increasingly evident that neither school of models alone is sufficient in providing fully satisfactory explanations to the empirical data. From the perspective of engineering modeling, the single-channel assumptions have thus far enjoyed a greater success, as indicated by the existence of a set of well-established models such as the queueing theoretic models and the network models reviewed in the following section. These models provide formal mechanisms for representing and codifying the single-channel assumptions of task selection in engineering terms. The multiple-resource models, in contrast, have only recently started to see some of their concerns being gradually accommodated in several simulation models of human performance, and there is still a lack of a set of computational methods to represent the assumptions of simultaneous execution and resource allocation in engineering terms. Furthermore, there does not exist a set of integrated engineering-based methods to model the concerns of both schools of models and to bridge the gap between the two. As indicated in the recent National Research Council report, there is a lack of methods to model the two most important features of a macromodel: task selection and simultaneous execution [3].

Manuscript received December 11, 1994; revised May 11, 1996. An earlier version of this paper was presented at the 1994 IEEE International Conference on Systems, Man, and Cybernetics, San Antonio, TX, October 1994, and published in the conference proceedings.

Y. Liu is with the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: yililiu@umich.edu).

Publisher Item Identifier S 1083-4427(97)00138-0.

Relationship between QN and Task Network Modeling

196 IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 27, NO. 2, MARCH 1997

Recently, Liu [29], [30] proposed that queueing network models and related methods employed widely in industrial engineering and system performance analysis may provide us an integrated computational framework for modeling the complex structural and temporal arrangements that multiple tasks might assume. The structural arrangements include both serial selection and parallel execution, and the temporal arrangements include both immediate activities and delayed processing. The purpose of this article is to examine the gap between the single-channel and the multiple-resource models of multitask performance, to describe a three-node queueing network model of interference between concurrent spatial and verbal tasks, and to illustrate the potential power of the queueing network approach for modeling multitask performance. As described below, the queueing network model provides a computational framework to integrate the concerns of the single-channel and the multiple-resource models, which, in essence, can be treated as special cases of the queueing network model.

The structure of the article is as follows. The first two sections review the single-channel and the multiple-resource models in detail, in terms of their theoretical assumptions and their applications in engineering modeling. Both the successes and the limitations of the two modeling approaches will be discussed. Then, a specific pattern of task interference between concurrent spatial and verbal tasks will be discussed as an illustration of the gap between the two classes of models. A three-node queueing network model is then described as a plausible and intuitive account of interference between spatial and verbal tasks and as an attempt to bridge the gap between the single-channel and the multiple-resource models. The value and the potential power of using queueing network methods to integrate some other currently isolated concepts of human performance and in modeling human-computer systems in general is discussed at the end of the article.

II. Single-Channel, Queueing Theoretic Models of Multitask Performance

As mentioned above, a prominent theory of multitask performance is the single-channel theory of selective attention. Its roots can be traced to the single-channel theory of human information processing originally proposed by Craik [12] to explain the psychological refractory period in human information processing discovered by Telford [51]. Telford discovered that when two reaction time tasks are presented close together in time, the reaction time to the second task stimulus is consistently delayed from a single task control condition. Various forms of single-channel theories have been subsequently proposed and elaborated [6], [14], [35], [56]. What is consistent about these theories is their common assumption that the human information processing system has bottlenecks that can only process one stimulus or piece of information at a time, and thus the system functions through a series of selections about which stimulus or piece of information to process. The focus of investigation is the identification of the bottlenecks, and the topics of debate among these theories are their different opinions regarding the locus of the bottlenecks and the factors that influence the selection processes.

The single-channel psychological theory of selective attention has been the fundamental basis of numerous engineering models of human performance [9], [42], [45]. These engineering models postulate that the human is a single-channel processor or a time-shared computer with a single central processing unit (CPU), which quickly switches and allocates its processing capacity among a variety of tasks in a sequential and all-or-none fashion. The models view human multitask performance as a single server queueing problem or multitask sequencing problem in which multiple tasks or diverse sources of information are queued for service from the single-server human information processing system.

Early models in this tradition have focused on modeling human visual sampling and monitoring behavior. Senders developed an instrument monitoring model, which integrated the single-channel concept and the sampling theorem of Shannon's information theory in making its predictions about the observer's fractional dwell time on each monitored instrument [45]. Carbonell proposed a single server priority queueing model of multi-instrument visual sampling [9]. The priority of each instrument at any instant is modeled as the combined effect of both the probability and the cost of exceeding a prescribed limit. The model integrates concepts from queueing theory, information theory, and decision theory. Carbonell used simulation to solve the model, and showed a close fit between the model's predictions and the subjects' actual performance in flying a simulator in terms of the fraction of attention devoted to each instrument. Senders and Posner further developed the queuing theoretic approach to instrument monitoring and provided analytical solutions to a model that they developed for display sampling [46]. Schmidt applied queueing theoretic method to the analysis of an air traffic control task [43].

A systematic and extensive effort in applying the queuing theoretic methods to the modeling of human machine systems can be found in a series of studies conducted by Rouse and his colleagues. Rouse described human–computer interaction as a queueing system with the human and the computer as two servers [42]. He formulated a queueing theoretic model of dynamic allocation of responsibility between the human and the computer in multitask situations, and illustrated the potential utility of this model with simulation experiments. Chu and Rouse later investigated the predictive power of the model with a behavioral experiment that simulated a multitask flight management situation [10]. A similar task scenario was also used in an earlier study by Walden and Rouse that investigated the suitability of a single-server queueing model of pilot decision-making [55]. Greenstein and Rouse integrated a pattern recognition technique called discriminant analysis with queueing theory methods in their two-stage model of human decision making in multiprocess monitoring situations [19]. Discriminant analysis was used in the first stage to generate estimates of event occurrence probability, and queueing theory was then applied in the second stage to incorporate these probabilities into the solution of the attention allocation problem.

Relationship between QN and Task Network Modeling

The single-channel assumptions can also be found in several other engineering models. For example, Sheridan assumed that there is a mental cost to switching attention which will determine how often different information sources in the environment are sampled, and thereby influence the sampling behavior [47]. Kleinman and Curry developed a model for human operator display monitoring, which assumed that the human is a single-channel time-shared processing channel [26]. Tulga modeled the multitask attention allocation problem as a dynamic single-machine sequencing problem [53]. The concepts and assumptions of single-channel processing and sequencing were also employed in the semi-Markov dynamic decision model of human task selection performance proposed by Pattipati, Kleinman, and Ephrath [40].

The single-channel concepts have been the fundamental basis of numerous simulation models of human performance. Notable examples include the task network models PROCRU [4], [28] and HOS [57]. Starting with the systems analysis of integrated networks of tasks (SAINT) modeling methodology developed by Siegal and Wolf [48], the task network approach models the human interaction with the environment as a sequence of tasks (also called paths), and acknowledges the existence of alternative paths to accomplish a goal or different goals in certain circumstances. These alternative paths form a task network. Parallel paths in a task network represent alternatives rather than concurrence of processing. Furthermore, a task in a network cannot be started until the preceding task on the same path of the network has been completed. Thus, at any instant only one task on a path can be executed.

Procedure-oriented crew model (PROCRU) is a control theory-based simulation model that has received widespread recognition [4]. The model is a closed-loop system model incorporating submodels for the aircraft, aircraft crew members, and the air traffic controller. The crew member submodel is a detailed human performance model and has a comprehensive coverage of human activities in monitoring and control of a large system. The model assumes that the crew members have a set of procedures or tasks to perform, and one task is chosen at a given instant of time, which is the one perceived to have the highest expected gain for execution at that time. The model contains a procedure selector, which is responsible for task selection and sequencing.

The human operator simulator (HOS) uses a library of human performance micromodels to simulate the operator's perceptual, cognitive and motor responses [21], [50], [57]. The original versions of the HOS approach assumes that humans are single-channel processors, and that human behavior is goal-oriented and can be defined as a sequence of discrete micro-tasks, which can be aggregated to predict task performance. Other simulation models of complex task performance that are based on single-channel assumptions include STALL [11], SIMWAM [17], [33], the model developed by Tulga and Sheridan [54], and its subsequent modified versions (e.g., [37]).

In spite of their differences, one of the common features of these models is their reliance on the fundamental assumption that humans can only process one piece of information at a time. Human multitask performance is modeled as a process of selecting tasks for sequential action according to some service discipline or cost function, which is usually based on the assumption that there is a mental cost to switching attention and/or there is a cost of being unable to attend to a critical instrument in a timely fashion [39], [42], [47]. Another common characteristic of these models is their focus on time as the underlying dimension and the metric of processing. Time is what is competed for by multiple tasks in a serial fashion and completion time defines the difficulty or demand of each task or task component. The models are relatively silent as to the intensity aspects of task demand.

These single-channel based models have demonstrated tremendous success in modeling two aspects of human performance: visual sampling in process monitoring and strategic task scheduling in high workload situations. The primary concern of visual sampling is to find the optimal tactics for a single-channel sampler to sample sources of information sequentially when the information sources cannot be attended to at once and thus compete for the operator's focal attention [36]. For example, in monitoring instrument panels, owing to the need for foveal vision when accurate reading is required, the eyes must be pointed in an appropriate direction to scan and sample a source of information. Since the distances between instruments are most often greater than the radius of foveal vision (2° to 3° of visual angle), accurate reading of spatially separated instruments can only be done sequentially. Similarly, in high workload situations when time pressure is the major source of workload and the operators are free to choose the order in which the tasks should be done to avoid overload [1], [37], these single-channel models have been quite successful in capturing the strategic scheduling aspect of task performance.

However, intuition and experimental evidence both support the view that humans do have the ability to perform multiple tasks in a truly concurrent fashion under many real-world circumstances. "We must recognize that people in fact can do more than one thing at a time and normally do" [1, p. 5]. In these task situations, single-channel assumptions and the analogy of a single-CPU time-shared computer or a single-server queueing system may only capture part of the nature of human performance and may not be adequate to portray the complex cognitive mechanisms for concurrent processing. Furthermore, each of the concurrent tasks may have attentional demands varying in intensity as well as in time. Some task pairs may be more similar with each other in their task structure than with others, and thus produce different patterns of task interference when they are performed concurrently with each other than when they are with other tasks. Thus, it may be necessary to address the parallel processing aspect of performance, to consider the intensity as well as the time characteristics of task demand, and to analyze the structural aspects of concurrent tasks. These issues have been the focus of investigation of models of divided attention.

III. Multiple Resource, Parallel Processing Models of Multitask Performance

In contrast to the single-channel assumption adopted by selective attention theorists that attention capacity can only