Creativity by Design: A Character Based Approach to Creating Creative Play

Joanna Bryson
Department of Psychology, University of Edinburgh, UK
joannab@dai.ed.ac.uk

Abstract

Creative play requires a fertile but well-defined design space in which to create. This paper describes one possible design process for creating virtual reality play spaces. The methodology is centered on designing AI characters for a *constructive narrative*. Requirements for the characters' agent architecture, and a three-layer design process for producing fertile and æsthetic narratives are described. This paper also discusses lessons learned from participating in a corporate virtual reality research and development effort.

1 Introduction

Creativity is generally agreed to be a Darwinian process involving novel recombination of existing design elements (Simonton, 1997; Boden, 1987). Part of what makes creativity challenging in artificial intelligence is that it often works on multiple levels simultaneously, casting a single element in two disparate roles. This is a problem not only for those planning AI representations, but also for those employed in designing props for encouraging creative play. The problem of designing creative play is to create a rich and interesting design space without limiting the creative potential of participants' experience in that space. This problem echoes the problem of flexibility vs. reliability in both natural and artificial learning systems (McGonigle and Chalmers, 1996).

One solution to this quandary is to allow the players to become constructors of their own experience. This idea has been explored to a much greater extent spatially than socially. Examples here include Internet MUDs, games such as SimCity, and constructive toys. Socially constructive toys also exist, though in fewer variants. Role-playing games, Purple Moon's 'girl games', where the objective is to find a place for the main character in an established society, and the game Creatures, which allows a player to evolve a society, are some examples. However, none of these allow the player to freely create characters and narratives of complex personal interactions. "Like good improvisational theater, cyberspace presents the opportunity for the audience to create its own characters and worlds, to write its own plots and stories, and to essentially become the directors, producers, and actors within their own imaginary worlds." (Pearce, 1997, pp. 345) For this potential to be realized, there must be tools for the creative design of personalities.

This paper describes a design approach for constructive narratives, and a design process for creating a creative play environment using this approach. The design process is separated into three components: a high level, highly artistic design element for creating story and characters; a middle, behavior based design level for creating personality in the agents; and a low level for design of basic behaviors. The middle level, the architectural approach for designing the characters, is particularly critical since it both facilitates and constrains the other two levels. The next section focuses on this level.

2 A Character Architecture for Constructive Narrative

Much research into agents for entertainment concentrates on the problem of combining the concept of a script with the notion of autonomous, reactive characters (Hayes-Roth and van Gent, 1997; Lester and Stone, 1997; André et al., 1998). The constructive narrative approach eliminates this problem by changing the top level creative design from a script to a cast of characters. Removing the script has the advantage of simplifying the addition, substitution, alteration, or removal of characters by the player. It has the penalty of removing a substantial element of structure from the palette of the initial creative designer: time ordered events. This problem has already been addressed by the creators of role-playing and adventure games. Structure is produced through the use of geographic space as well as character personalities. Plot, if desired, can be advanced by knowledgeable characters, found objects, and revealed locations. Personality traits such as loyalty and agrophobia can be used to maintain order despite the presence of a large number of autonomous characters.

Most virtual reality agent architectures are fundamentally behavior based, and at least partially reactive (see Sengers, 1998, for a recent review and critique). This is because the reactive, behavior based AI revolution of the late 1980s was primarily the triumph of a design approach. Behavior based AI is simpler to design than a monolithic intelligence system simply because it requires the decomposition of intelligent behavior into easy to program modules. Specifying that the intelligence should also be reactive removes the complex problems of learning and constructive planning from the agent. Unfortunately, it simultaneously limits the potential complexity of the agent's behavior. Nevertheless, by empowering human designers, the behavior based approach has been more successful than any fully human-specified or fully machine-learned approach.

The limitations of completely reactive systems have been widely recognized, and are addressed in numerous architectures (see for example Hexmoor, 1997). Some authors have proposed that the community has moved beyond both constructive and reactive planning to a new dominant paradigm, situated planning Levison (1996); Kortenkamp et al. (1998). Situated planning architectures generally include reactive behaviors, pre-stored plans, elements of learning, and possibly constrained forms of online planning. Two of the most popular architectures of this paradigm are PRS (Georgeff and Lansky, 1987) and 3T (Bonasso et al., 1997). Both of these have at their center a scripting language for allowing the specification of sequential and hierarchical behavior structures. These structures provide additional information (in the form of internal state) for action selection in situations that might be perceptually identical. This allows the situated planner greater behavioral flexibility than the fully reactive planner, which is dependent on current sensing to select its next action. The script structures also allow for the combination of simple behavior elements into larger modular forms, again simplifying the design task.

The work described below uses a less well established architecture, Edmund (Bryson and McGonigle, 1998). The principle advantage of Edmund over the two architectures mentioned above is that Edmund better maintains and develops the behavior based concept, particularly with regard to specialized perception. The behavior based ideal is for perception and action to be inexorably linked through the behaviors (Brooks, 1991; Matarić, 1997). Another principle is that perception should be specialized to the task (Horswill, 1993; Hallam and Malcolm, 1994). Edmund extends these principles with the observation from the natural sciences that sophisticated perception always requires memory (Pöppel, 1994; Barlow, 1994; von der Malsburg, 1995, e.g.). Behaviors in Edmund are objects, in the software engineering sense of the term. The state of the object serves as specialized perceptual memory. The primitive action and sensing elements referred to by Edmund's scripting language are methods on these behavior objects. Perceptual memory is necessary for many types of perceptual discrimination, as some categories can only be perceived by combining instances of sensory information received over a period of time. Having behavior linked with memory also allows for longer-term learning and other persistent internal states, such as emotions.

Edmund's scripting language allows for four levels of control structure. The most basic level is composed of action and sensing primitives, which interface directly to the behaviors. Next there are two fundamental sorts of conglomerates. The first is an *action pattern*, a simple sequence which runs uninterrupted unless a sense-predicate element indicates radical failure. The second is a *competence*. A competence consists of a prioritized set of elements, whose behavior tends to converge towards the highest priority element, the goal. When a competences is active, it selects the highest priority element which is currently capable of being executed. If that element is the goal the competence finishes successfully, if no elements fire the competence fails. Competences are similar to teleo-reactive plans (Nilsson, 1994), and other reactive planning mechanisms. Their elements may consist of either behavior primitives, action patterns or other competences.

The highest possible level of an Edmund script is a special form of competence called a *drive*. The elements of a drive provide the activation or motivation for a coherent section of the script. A drive's elements, while prioritized like a competence, may also be scheduled. Thus if a high priority element has fired recently, it may be inhibited in order to allow lower priority elements to execute. Drives also maintain overall behavior coherence by being persistent. As described above, competences and action patterns both terminate routinely. Further if a competence selects an element which is itself a competence, the parent competence is replaced by its child in the action scheduling. This feature is called a *slip stack hierarchy*, and allows for indefinite behavior chaining or looping. A drive remembers its initial set of elements, so that if one element terminates it is replaced by the original element. In the case of chaining competences, this allows an Edmund script to 'pop stack' reactively. After a competence terminates, each previous competence of that competence's controlling drive element chain is revisited in order. If the situation has not changed the parent of the terminating competence will be reached, if the situation has changed a more appropriate behavior may be selected. In the meantime, while a particular competence is excuting there is no bottleneck of a long stack to be checked before each individual action.

Edmund's scripting structure differs from 3T's by allowing indefinite chaining of subelements. It differs from PRS by having no external manipulation of the control structure.

For the purpose of the project described below, Edmund has been combined with another character architecture, Ymir (Thórisson, 1999), into a new hybrid architecture called Spark of Life or SoL. Ymir is designed to support multimodal dialogs between human players and artificial characters. It includes a complex scheduling and prioritization system for handling verbal and postural perceptual information, and for producing verbal, gesture and facial expression output, both in real time.

SoL's three most important extentions of Edmund's capabilities gained from Ymir are:

- an extensive encapsulated knowledge of psychosocial data for creating believable and relevant interactions between humans and animated agents,
- a system for selecting between multiple possible expressions of a particular behavior by choosing the one most appropriate given the current physical configuration of the agent, and
- a cerebellum-like system for moving the agent from the current configuration to the next chosen one.

The development of SoL will be described in detail in (Thórisson and Bryson, In preperation). The remainder of this paper discusses the process of designing a constructive narrative. The next section returns to the description of the design process, providing a task decomposition. This is in turn followed by a description of personal experience working with this methodology.

3 Designing Agents for Creative Play

As mentioned in the introduction, creative play consists principally of the novel recombination of established elements. In fact, the evolutionary utility of play is considered to lie in enabling an individual to acquire and rehearse complex behaviors, as well as to learn appropriate situations in which to express them (Bekoff and Byers, 1998; Byrne and Russon, forthcoming). In the relatively pragmatic and demanding field of entertainment, it would be a mistake to attempt to design agents for creative play that were expected to be as self-sufficient as children in developing such skills. Even were the designers' skills and knowledge up to such a task, children themselves take years to acquire such behaviors to any degree of entertaining proficiency.

Similarly, AI developers should not necessarily be expected to be sufficiently skilled artists that they can create the plots and characters necessary for a fully engaging interactive play experience. AI seems to attract (possibly even to require) developers with a hubristic belief in their own ability to replicate the thinking skills of others. However, good artists devote years of attention, and sometimes formal education, to perceiving and constructing the things that make a situation interesting, æsthetic and fun. The following design process places the AI developer as an intermediary between the artistic and the engineering aspects of the project. The AI developer is in the best situation to understand both requirements and restrictions of the overall project, and therefor has considerable responsibility for communication as well as developing solutions.

As a developer, the AI expert is responsible for taking a set of motivations, goals, knowledge, personality quirks and skills, and creating an agent that will behave reasonably rationally. The character should be able to prioritize its goals and display its intentions. It should exhibit both persistence and resolution while at the same time being aware and opportunistic. In short, it should have a recognizable personality. Developing the initial set of character attributes, however, is not necessarily solely the task of the expert in agent development. It *is* necessarily the task of one or more creative artists. The artist's responsibility is to provide well formed and interesting characters, skills and situations, to design potential plots and plot twists. In this, as in most industrial design, it will be best if the artists work in a team with the agent developers, who can help the artists understand the limits of the agent's behavioral and expressive capabilities.

The agent developers are themselves constrained by the platform on which the artificial agent is to be expressed. In virtual reality, these constraints are provided by the graphics environment in which the agent will be designed; in robotics, they are provided by the robots. It is the responsibility of the AI developer to provide requirements for, and understand the constraints of, the underlying platform — just as the narrative developer must understand the capabilities of the agents. Again, the character personality developer may or may not be the correct person to develop the agent's behavioral platform. This does not apply only to 'technical details' because the platform in this context may also provide the basic behaviors, or behavior primitives, for the agents. In this case, the platform developers are also responsible for artistic input to the project, as they need to create believable and attractive behavior environments.

The design process should obviously happened iteratively. Many forms of technical constraint might only be recognized after development has begun. Further, as the system develops, it can provide considerable creative inspiration to the designers. Even more importantly, early users, particularly those coming from outside the project, will discover both shortcomings and unforseen creative potential in the system. All of these sources of information should lead to periods of redesign and renegotiation between the various levels of the project.

4 Case Study: Creating Characters for an Adventure Narrative

The design process described above was developed as part of a blue-sky research effort to create an interactive virtual reality entertainment package that allows a child to engage in creative and constructive play inside the framework of an established action/adventure environment. The funders of this research have given permission for this account, but have asked that they not be identified in print. The project has not been brought to full product; consequently, this paper can only report partial results. However, progress has been sufficient that this case can be used to illustrate the design principles above, and to give some indication of the efforts and difficulties involved. For the purpose of this paper, this project will be called "the castle character project". This phrase will be used to refer to the AI portion of a large-scale, multifaceted research effort.

4.1 High Level Design

In the case of the castle character project, much of the creative environment was predetermined, as it was a virtual version of an active product. Consequently, the general appearance of the characters and their world, and an outline of the characters' personalities, had already been developed. The domain was a magic castle, inhabited by an evil knight and various magical entities. Much of the overall research effort was dedicated to ensuring that simply exploring the space would be intrinsically rewarding. However, this paper will focus on the subproblem of providing fun and interest through interaction with intelligent characters.

The first step to creating and interesting narrative for a set of characters is to understand the constraints of the task and the system. One set of constraints is determined by the technical specifications of the character's environment. In the castle character project, such constraints included the fact that open spaces within the castle were not significantly larger than the characters themselves. Also for technical reasons, the characters had limited flexibility, which constrained their movements and their gestures. Speech recognition was also technically difficult and unreliable. Consequently only one character, who was visibly fixed in place, was chosen for verbal interactions. These interactions were expected to be in the form of questions and answers, so that the character need only recognize a fixed set of queries or requests.

The next set of constraints are those dependent on the expected users of the system. Because the users of the castle character project were expected to be naïve to VR and only exposed to the system for a few minutes, it was considered essential that the characters provide interest whether or not the user deliberately attempted to interact with them. Thus, the characters should interact with each other. They should also react to the visitor in their domain in a way that encouraged exploration, but they should not be too forceful or too intrusive on the user's experience. To maintain interest, the characters should act and interact in such a way that they will generate continuous change. There should be no steady state that the system of characters can reach if the user is being passive.

Because of the constraints mentioned in the previous paragraphs, most of this change had to take the form of simple arrivals and departures, as well as a few gross gestures. This effect was achieved by designing characters with various incompatible goals. For example, a witch could frequently fly around the castle in a quest for intruders. However, when she found the intruder, she could do little other than land near and slightly approach the stranger, and cackle. However, her presence might attract other characters, some of whom might in turn repulse her. By having characters that are attracted by some situations, yet repulsed by either crowds or other characters, the number of simultaneous interactions, and therefor the amount of confusion, can be limited. Also, the amount of free space for character motion can be maintained.

4.2 Encoding Personality

Starting from the descriptions of the characters set by the marketing department of the product, and keeping in mind the constraints determined in evaluating the task, each character was described in terms of three to five goals or drives. Further, the behavior associated with achievement of these goals was visually described. This work was done by a team of in-house artists and external creative consultants, with the AI team participating both creatively and as technically informed resources.

Once the personality of the characters has been sketched, the next step is to code it. Under the Edmund approach described earlier, this involves developing first behavior libraries, and second control scripts. The behavior libraries provide the sensory and action primitives for the system, while the control scripts can be thought of as reactive plans. This process consists of:

- Prioritizing goals or gross behaviors and determining their necessary preconditions. For example, the witch described above has a goal of patrolling the castle from the air. This has a fairly high priority, but the motivation should be reduced by the performance of the act, so that in general she circles the castle only three times. She also has a priority of landing in a room in which she has seen an intruder, once she no longer desires to fly. She also avoids bats.

- Determining behavior primitives and behavior state necessary. For example, the witch has to remember if she saw an intruder on her patrol. Another example: a bat might approach an intruder closer and closer over successive swoops, but back off if the intruder waves their arms. This would require a piece of state within the bats swooping behavior showing how bold it is feeling in order to determine its trajectory. Some characters might be made into friends by playing with them. These would have to remember how friendly they feel towards a particular person. Seeing the user, avoiding the walls of the castle, flying and landing are behavior primitives required by all of these agents.

- Developing and testing the behavior libraries and the scripts.

4.3 Developing Behavior Primitives

In developing behavior libraries, the task of the personality designer connects to the task of environment's architects. For the castle character project, some of the potential difficulties of this relationship were overlooked, and caused some of the greatest difficulties of the AI effort.

There are several possible approaches for building the basic behaviors. One straightforward approach would be for the character developers to program the behaviors from scratch using models prepared by the graphic artists. There is a general problem for this approach: as mentioned earlier, AI programmers are not necessarily artists or students of natural motion. Animals have evolved complex motion behaviors, constrained by physical forces and structures not normally modeled on an artifact, particularly one designed to run in real time, so difficult to take into account. Animals are also constrained by habits of behavior, whether general to a species or specific to an individual. Even if æsthetic motion primitives are achieved by an AI programmer, the process of programming them is likely to have been very time-consuming.

Besides this general problem, the castle project also ran into an avoidable problem. The AI programmers took for granted an agent-oriented view within the graphical environment. That is, we anticipated being able to direct any element forwards or backwards, right or left, up or down, and at a particular speed relative to the rest of the world. However, the graphic artists and modelers were used to working only for the perspective of the camera. Consequently, coordinate frames and even size/distance metrics were not always consistent between the various models. This led to obvious problems in developing allocentric motion routines.

Another potential source of behavior primitives explored on the castle character project were the efforts of a team of animators already working on the project. Animators are trained artists who create many lifelike behaviors in the course of an animation. The idea was to segment animations into sets of behaviors suitable as multiple exemplars of various behavior primitives. As mentioned earlier, the Ymir basis of SoL would be able to select an appropriate instance from such a set and move the character smoothly to that instance. Thus a continuous variety of behavior could be derived from combining and connecting fixed sets of 'canned' behavior. Unfortunately, the animations proved as slow and difficult to develop as the hand-programmed routines. More importantly, the format the animations were produced in was determined to be incompatible with the primary real-time virtual reality environment.

One successful strategy was eventually found: a purpose built animation tool for "quick and dirty" animation segments stored in an appropriate format for the main VR engine. Motion capture is another possible source of natural looking behavior primitives, but it has not yet been explored for this purpose on the castle character project.

5 Discussion and Conclusions

Creativity is analogous to learning. In both processes, something is built by the agent's actions over time. In the case of learning, these actions are often considered implicit, built-in techniques, and the thing changed is usually the agent's own knowledge. In creativity, the actions are more often expected to be explicitly represented learned skills, and the state changed is expected to be external to the agent, so that it can be appreciated by others. However, the lines between these two behaviors are not so clearly drawn. Clark (1996) suggests that the construction of an agent's learned knowledge, particularly of a person's, consists of both memories and artifacts; while human fantasies are creative constructions that are usually completely internal. Skills for learning can be learned, and talent for arts can be inborn.

A constructive narrative is therefor creative on several levels. In creating a creative experience, the goal is to provide both interesting media for expressing the content to be recombined, and tools that facilitate the recombination. If the media provided also includes active creators — for example, agents that autonomously create situations and social dynamics — then the user has the opportunity for highly complex production. This kind of creative experience is currently only afforded to people such as composers and writers of drama, corporate managers and public policy makers. Designing such a system can in itself be a highly creative act, but it is particularly challenging to do so in such a way as to allow the users ample opportunity to express their own creativity. Of course, in the commercial climate, there are often intentional constraints to orient users towards certain forms of creative elements that exploit particular products, but these are not necessarily a burden. People like to create within well-understood spaces and forms: creativity is not chaos.

A creative environment with constantly changing stories and adventures can be developed by using artificial intelligence and design techniques that exploit and express the creativity of the designers. The intelligent agents in these environments are literally agents of creativity rather than being significant creators themselves: they embody the rules and knowledge both invented and learned by their own creators. This paper has presented a design approach for creative environments called constructive narratives, and a design process for creating a creative play space under this approach. The design process focuses on the roles of the various team members in communicating and constructing an interesting reality based around AI characters. The characters are implemented using behavior based techniques, for simplicity of design, combined with situated planning devices, to allow for complexity of characterization and behavior. We have described our experiences in using this process. This work is still in progress — we hope to eventually develop more fully interactive characters, and more open narrative architectures that allow the users to design characters as well.

Acknowledgements

The team who created the project and the environment in the case study above were: Dent-de-Lion du Midi, Christian Greuel, Svend Tang-Petersen, Victor Bonilla, Dan Mapes, Claude Aebersold, Inge E. Henriksen, Preben Kæseler, Kristinn R. Thórisson, and Julian E. Gómez. Their work as artists, visionaries and engineers is gratefully acknowledged. The AI effort was headed by Kristinn R. Thórisson, and included in addition to those mentioned above, Mikkel Arentoft, Michael Thompsen, Henrik Bach Ravn and Randahl Fink Isaksen. Lieselotte van Leeuwen assisted with child psychology; Celia Pearce assisted with concept development; Jacob Buck, Súsanna Thorvaldsdóttir, Michael Nielsen, and Christian George helped with technical implementation. The castle project was developed using Silicon Graphics equipment and expertise. Thanks also to Will Lowe and the anonymous reviewers and lawyers who contributed to the final form of this paper.

References

Elisabeth André, Thomas Rist, and Jochen Müller. Integrating reactive and scripted behaviors in a life-like presentation agent. In Katia P Sycara and Michael Wooldridge, editors, *Proceedings of the Second International Conference on Autonomous Agents*, pages 261–268. ACM Press, 1998.

Horrace Barlow. What is computational goal of the neocortex? In C. Koch and J. L. Davis, editors, *Large-Scale Neuronal Theories Of The Brain*, pages 1–22. mitpress, 1994.

Marc Bekoff and John Alexander Byers, editors. *Animal Play: Evolutionary, Comparative, and Ecological Perspectives*. Cambridge University Press, 1998.

Margert Boden. *Artificial Intelligence and Natural Man*. Basic Books, New York, 2nd edition, 1987.

R. P. Bonasso, R. J. Firby, E. Gat, D. Kortenkamp, D. P. Miller, and M. G. Slack. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental & Theoretical Artificial Intelligence*, 9(2/3): 237–256, 1997.

R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

Joanna Bryson and Brendan McGonigle. Agent architecture as object oriented design. In Munindar P. Singh, Anand S. Rao, and Michael J. Wooldridge, editors, *The Fourth International Workshop on Agent Theories, Architectures, and Languages (ATAL97)*. Springer-Verlag, 1998.

Richard W. Byrne and Anne E. Russon. Learning by imitation: a hierarchical approach. *Brain and Behavioral Sciences*, forthcoming.

A. Clark. *Being There: Putting Brain, Body and World Together Again*. MIT Press, Cambridge, MA, 1996.

M. P. Georgeff and A. L. Lansky. Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 677–682, Seattle, WA, 1987.

John C. T. Hallam and Chris A. Malcolm. Behaviour, perception, and action — the view from situated robotics. *Philosphical Transactions of the Royal Society of London*, 349:29–42, 1994.

Barbara Hayes-Roth and Robert van Gent. Story-making with improvisational puppets. In W. Lewis Johnson, editor, *Proceedings of the First International Conference on Autonomous Agents*, pages 1–7. ACM press, February 1997.

Henry Hexmoor. Special issue: Software architectures for hardware agents. *Journal of Experimental & Theoretical Artificial Intelligence*, 9(2/3), 1997.

Ian D. Horswill. *Specialization of Perceptual Processes*. PhD thesis, MIT, Department of EECS, Cambridge, MA, May 1993.

David Kortenkamp, R. Peter Bonasso, and Robin Murphy, editors. *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems*. MIT Press, Cambridge, MA, 1998.

James C. Lester and Brian A. Stone. Increasing believability in animated pedagogical agents. In W. Lewis Johnson, editor, *Proceedings of the First International Conference on Autonomous Agents*, pages 16–21. ACM press, February 1997.

Libby Levison. *Connecting Planning and Action via Object-Specific Reasoning*. PhD thesis, University of Pennsylvania, March 1996. School of Engineering and Applied Science.

Maja J. Matarić. Behavior-based control: examples from navigation, learning, and group behavior. *Journal of Experimental & Theoretical Artificial Intelligence*, 9 (2/3):323–336, 1997.

Brendan McGonigle and Margaret Chalmers. The ontology of order. In Les Smith, editor, *Piaget: A Critical Assessment*. Routledge, 1996.

Nils Nilsson. Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1:139–158, 1994.

Celia Pearce. *The Interactive Book : A Guide to the Interactive Revolution*. Macmillan Technical Publishing, 1997.

E. Pöppel. Temporal mechanisms in perception. *International Review of Neurobiology*, 37:185–202, 1994.

Phoebe Sengers. Do the thing right: An architecture for action expression. In Katia P Sycara and Michael Wooldridge, editors, *Proceedings of the Second International Conference on Autonomous Agents*, pages 24–31. ACM Press, 1998.

Dean Keith Simonton. Creative productivity: A predictive and explanatory model of career trajectories and landmarks. *Psychological Review*, 104:60–89, 1997.

Kristinn R. Thórisson. A mind model for multimodal communicative creatures & humanoids. *International Journal of Applied Artificial Intelligence*, 1999.

Kristinn R. Thórisson and Joanna Bryson. Flexible behavior-based planning for multimodal characters capable of task-oriented dialogue and action. In preperation.

Christoph von der Malsburg. Binding in models of perception and brain function. *Current Opinion in Neurobiology*, 5:520–526, 1995.