

一个基于情景演算的自主非玩家角色模型研究

黄向阳¹, 彭 岩¹, 张树东¹, 涂序彦²

(1. 首都师范大学信息工程学院, 北京 100048; 2. 北京科技大学, 北京 100083)

摘 要: 与传统的将人工智能技术嵌入到游戏引擎的设计方式不同, 通过为自主非玩家角色 (Autonomous Non-player Characters, ANPCs) 设计一类智能控制器将人工智能技术从游戏引擎中分离出来. 用户通过三个接口与智能控制器交互, 三个接口分别描述了 ANPC 的感知、思考以及行动. 提出了一种扩展的情景演算来融合三个接口, 并解决 ANPC 行动结果的不确定性以及为 ANPC 从认知模型回到实用世界模型提供形式化. 基于 ANPC 模型实现了一个用于游戏的人工智能平台, 实验证实利用该平台可以很方便地创建与真实玩家对抗的 ANPCs.

关键词: 非玩家角色; 智能控制; 认知模型; 情景演算

中图分类号: TP18 **文献标识码:** A **文章编号:** 0372-2112 (2010) 05-1221-05

A Model for Autonomous Nonplayer Characters Based on Situation Calculus

HUANG Xiang-yang¹, PENG Yan¹, ZHANG Shu-dong¹, TU Xu-yan²

(1. Information Engineering School, Capital Normal University, Beijing 100048, China;

2. University of Science and Technology Beijing, Beijing 100083, China)

Abstract: Historically, nonplayer characters (NPCs) were simulated as part of the game logic. Unlike traditional techniques, a new intelligent controller (IC) was designed for autonomous NPCs (ANPCs), which separates AI logic from game engine. Users communicate with the IC by three interfaces which define sensing, thinking and acting of ANPCs. An extension to situation calculus is proposed to help implement the IC. The extension includes the introduction of epistemic fluent and the definition of linguistic epistemic fluent to solve uncertain effects of actions and the potentially uncountable number of possible worlds, and incorporating sensing into the effect axioms to make ANPC come back the real-world model from its cognitive model. A game AI platform based on the model is built. It proves to be especially convenient for creating ANPCs which can counteract with player characters by applying the platform to game Quake 2.

Key words: nonplayer character; intelligent control; cognitive model; situation calculus

1 引言

随着硬件技术的迅猛发展、3D 技术的日臻完善, 计算机游戏中的人工智能技术逐渐成为决定一个游戏成败的关键因素^[1,2]. 近年来, 在游戏中创建了许多极具个性的智能体 (在游戏中称为 Nonplayer Characters, NPCs), 如游戏 *Creatures* 中具有心理和生理状态的 Norns 精灵、游戏 *Black & White* 中具有学习能力的精灵、以及游戏 *The Sims* 中使用人工生命技术创建的个性精灵等. 这些具有部分自主能力的智能体增加了游戏的可玩性. 然而在这些游戏中, 人工智能技术只是作为游戏逻辑的一部分来研究, NPC 仅仅是一段“聪明”的程序. 现在许多人认为 NPCs 应该是完全自主且更具智能: 它们自己和虚拟世界交互; 用各种学习算法来适应环境^[2,3], 称

这样的 NPCs 为 ANPCs (Autonomous NPCs). 但是, 在游戏中创建像玩家 (Player Characters, PCs) 一样真实的自主角色, 也是游戏开发中最困难的方面之一.

另外, 在实现方面, 当前游戏人工智能技术开发者面临的挑战有^[2]: (1) 创建一个易于调整和控制的人工智能系统; (2) 找到好的系统设计模式和人工智能技术以便减少工作量. 现在的开发者很少有机会开发高层次的人工智能技术, 他们将花费大量时间在低层次的细节上, 如路径搜索^[2]. 一些研究者认为: 游戏人工智能技术的下一个突破口将依赖于人工智能中间件、组件技术以及联结它们的合适的接口^[1,2]. 为了能将人工智能很好地服务于游戏, FEAR 工程 (<http://fear.sf.net/>) 采用了一种灵活的架构来创建反应式仿生机器人, 人工智能接口标准协会 (AIISC, <http://www.igda.org/ai/>) 的“世界接口”, 试

图使得各种厂商的人工智能中间件能够集成起来。

目前还没有工作把实现 ANPC 的高级人工智能技术(特别是认知建模技术)和实现 ANPC 的软件架构联系起来.为此,本文尝试在情景演算基础上给出一个开放式的 ANPC 模型,游戏开发者只要有选择地实现一些接口中的功能便可生成一类新的 ANPC,这种模式可以将人工智能技术开发者从底层的细节中解放出来,而 ANPC 的自主性,则是通过基于情景演算的认知建模来实现.下面描述如何将实现架构并入到情景演算中。

2 ANPC 的架构

传统的 NPC 控制器只是作为游戏逻辑的一部分来设计.考虑只有一个角色的场景,场景的变化将只受控于该角色的输出,在线性定常的条件下,可以用下面的状态方程来描述游戏场景的状态变迁:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}\end{aligned}\quad (1)$$

式中, $\mathbf{x}(t)$ 为游戏场景状态矢量, \mathbf{A} 为对象矩阵, \mathbf{B} 为控制矩阵, \mathbf{C} 为观测矩阵, \mathbf{D} 为直控矩阵, $\mathbf{y}(t)$ 为游戏场景输出矢量,也是角色输入矢量. $\mathbf{u}(t)$ 为角色输出矢量,可写成: $\mathbf{u}(t) = \{u_1(t), u_2(t), \dots, u_n(t)\}$.

然而,实际中,游戏场景的状态变迁较(1)式复杂得多,难以用数学公式来表达,为此,用广义算子模型来描述^[4]:

$$Y(\omega) \doteq K(\cdot)U(\omega) \quad (2)$$

式中, \doteq 是一种广义的等号,它可以表示相等、蕴涵、以及近似等多种含义, $K(\cdot)$ 为广义算子,既可以是定性的知识推理过程的智能操作,也可以是定量的数学运算,广义输出 Y 是广义输入 U 经广义算子 $K(\cdot)$ 变换或传递而产生的结果.对于 ANPC 来说,游戏场景的变化规律是一个“黑箱”,只有通过不断地尝试和校正自己的行为才能达到目标,这就是目标导向行为(Goal-oriented Action, GOA).

为了设计具有 GOA 能力的 ANPC,以及提供一个更为灵活的人工智能实现模式,本文依据公式(2)为 ANPC 设计一类新的智能控制器.为了能够重用游戏引擎的工作,ANPC 的控制器将由传统控制器和智能控制器复合而成,原理如图 1 所示。

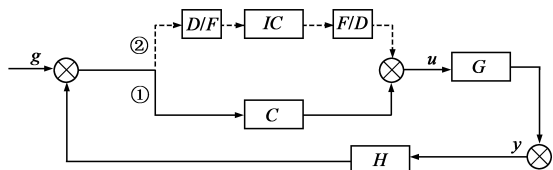


图1 由智能控制器和游戏引擎构成的复合控制器

图 1 中,有两个分支,实线分支为游戏引擎提供的控制器 C ,虚线分支为智能控制器 IC , D/F 为模糊感

知变换, F/D 为解清晰变换, G 为被控的游戏场景对象, H 为反馈传感通道, g 为 GOA 中的目标, u 为通过仲裁器控制的输出, y 为游戏场景输出反馈.仲裁器根据一定的规则决定两类控制器的合成输出。

下面来分析智能控制器的设计.从图 1 知道智能控制器的输入来源于游戏引擎,从游戏引擎到智能控制器的输入称为感知接口,简称感知,感知的形式化定义如下:

定义 1(感知的定义) W 是智能控制器关心的属性集合, W 非空, $W \subseteq y$, 将 W 映射到一个非空的语言变量集合 \mathcal{K} , 对于 \mathcal{K} 中的任一元素 ω , 在 y 中有唯一元素与之对应, ω_i 是 ω 的第 i 个模糊子集, μ_i 是 ω_i 的隶属度函数, 设 $P = |\mu|$, 表示 \mathcal{K} 中所有元素的隶属度函数构成的矩阵, 则 P 称为感知矩阵, 简称感知或感知接口.对 ω 的感知记为 P_ω .感知接口是一个函数的集合。

智能控制器的输出需要游戏引擎的执行,从智能控制器的输出到游戏引擎的渲染称为行动接口,简称行动,下面给出行动的形式化定义。

基本行动 α 是基本行动,当且仅当在公式(1)中的输出向量 u 中存在唯一元素与之对应,或者 α 为空操作。

定义 2(行动的定义) (1)基本行动是行动;(2) α 是行动,当且仅当 α 是有限次使用复合规则(包括序列、条件以及循环等^[1])得到的.用 E 表示所有行动的集合, E 也称为行动接口,简称行动。

行动接口是一个行动函数的集合,是智能控制器的输出与游戏引擎的接口,如果智能控制器输出的行动不包括在行动接口中,则以空操作处理.定义了感知和行动接口后,我们可以给出 ANPC 的实现架构。

定义 3(架构的定义) 架构是一种实现的框架,定义 ANPC 的架构为:

$$IMP_{ANPC} = E \circ U \circ P \quad (3)$$

其中 P 、 U 、 E 分别为感知接口,认知接口以及行动接口,分别简称为感知、思考以及行动,“ \circ ”为从右向左的传递操作. ANPC 通过对外界的感知,内部的思考,最后产生输出到外部游戏世界.模型通过 P 和 E 将游戏中的人工智能技术与游戏引擎隔离,使得游戏中的人工智能技术可以在一个独立的层面上研究。

3 认知模型

ANPC 关于它所生存和活动的虚拟世界有自己的内部模型,称为认知模型,与认知模型对应的是虚拟世界实际采用的模型,称之为实用世界模型. ANPC 的架构涉及的概念表明,情景演算^[1,5]正好适合用来实现 ANPC 的认知模型.由于文献[1]对情景演算用于计算

机游戏和动画作了比较系统的描述,因此本文对 ANPC 的认知建模继承该文献的工作,包括有关“老框”等问题的形式化.为了解决 ANPC 行动结果的不确定性以及认知模型的实用性,本文对情景演算进行了扩展,在情景演算中引入 K-流来表达行动结果的不确定性,并引入感知使得认知模型可以变换到实用世界模型.

3.1 情景演算(Situation calculus)

情景演算是用一阶谓词逻辑描述变化世界的方法.在情景 s 下,由于采取了行动 a ,于是得到情景 s' ,由函数 do 给出,可表示为 $s' = do(a, s)$.

前提条件公理(Precondition Axiom):前提条件公理对于行动 $a(x)$ 什么情况下可以发生,给出了充分必要条件,由谓词 $Poss$ 表示,如, $Poss(a(x), s)$.

效应公理(Effect Axiom):在采取了一个行动后,对于流所应取得的值,效应公理给出了必要依据.如, $Poss(shoot, s) \wedge Aimed(s) \Rightarrow \neg Alive(do(shoot, s))$ 表示如果瞄准了 NPC,并且能够射击,那么在射击后, NPC 就死了.

情景演算方法描述了一棵以初始情景 S_0 为根的情景树,每个分支都是采取一种行动的结果,但是对于一个有多种结果的行动,直接描述却不是很方便,为此本文引进 K-流来描述具有多种可能结果的行动.

3.2 K-流(Epistemic fluent K)^[1]

角色跟踪了解世界可能发生的情景的方法是去定义一种认知流 K ,这种 K-流跟踪了解所有与 K 相关的世界.因此,我们用 $K(s', s)$ 表示在情景 s 下,而角色可能认为是另一种情景 s' 的情形.这就是说,在角色的知识中情景 s 和 s' 是不能分辨的,只有通过感知后才能分辨是什么情景. K-流事实上是模态逻辑的“可能世界”观点.

在情景演算中利用认知流 K 将“知识”模型化.但是,还存在一些问题,其中一类问题与技术实现有关,即表达角色世界初始情景的困难.本文定义一类语言认知流来解决这个问题.

3.3 语言认知流

在情景演算中提供了一种完美的数学表达语言,但是,当我们遇到在实数域 R 上取值连续的函数流,事情将会变得很糟糕:我们不可能列出不计其数的“可能世界”.为此依据感知的定义(定义 1)来定义一类语言认知流(Linguistic Epistemic Fluent, LEF).

定义 4(语言认知流) 一个 LEF 流是一个三元组 $(\omega, T(\omega), \nu)$, 其中 ω 是流的名字, $T(\omega)$ (或简写 T) 是流的取值(语言值)的名字的集合,流的每一个取值是一个模糊变量,这些模糊变量定义在区间 ν 上,并且 ω 满足公式(4)的限制.

$$[\omega_1]_0 \cup [\omega_2]_0 \cup \dots = \cup [\omega]_0 = \nu \quad (4)$$

其中 $\omega_1, \omega_2, \dots$ 为流 ω 的所有取值, $[\omega]_0$ 为 ω 的 0-截集.下面以 K-流和 LEF 来表达 ANPC 行动的不确定性.

3.4 行动的不确定性

ANPC 所处的世界往往是高度复杂的,它的行动结果可能是不可预知的,尤其是在有真实玩家参与的情况下.例如,ANPC 向真实玩家开枪,结果可能打中,有可能被玩家躲开.情景演算理论上可以表达这种行动结果的不确定性.如,在行动和行动的可能结果之间插入一个中间情景来表达行动的不确定性^[6],但是这种方法在语意上显得很生硬. ANPC 认知模型使用 K-流来描述这种行动的不确定性,使得情景演算在形式上和语意上显得更为自然.

假设所关心的 LEF ω 存在 n 个值 $\omega_1, \dots, \omega_n$. 那么执行拥有不确定结果的动作 a 后,关于 K-流的效应公理表示如下:

$$\begin{aligned} Poss(a, s) &\Rightarrow (s' = do(a, s)) \\ \wedge [(K(s_1, s') \wedge \omega(s_1) = \omega_1) \\ \vee \dots \\ \vee (K(s_n, s') \wedge \omega(s_n) = \omega_n)] \end{aligned} \quad (5)$$

3.5 感知

当 ANPC 在采取某种行动时,事先并不能确知自己的行动是正确的,只有在采取了这种行动之后,再查看发生了什么效应,若结果不是所期望的,则进行某些修正,这个过程称为感知.将感知加入到情景演算的语法中使得认知模型可以转换到实用世界模型.

假设所关心的每个 ω 值的感知(ω 为 LEF),有一个对应的产生知识的行动 P_ω . 那么,关于 K-流的效应公理表示如下:

$$Poss(a, s) \wedge a = P_\omega \Rightarrow [\forall s_i (K(s_i, s) \wedge \omega(s_i) = \omega(s))] \quad (6)$$

上面的效应公理表达了所需的感知概念.在某个 s 情景下,如果角色采取了感知行动,其效应是对与 K 相关的情景进行约束,使其值与情景 s 下的值相符合,而并不会改变世界的值,从而当感知行动发生时,行动的不确定性将变得确定,认知模型可以变换到实用世界模型.

3.6 学习

对于 ANPC 来说虚拟世界正像一个“黑箱”,ANPC 需要对它进行学习. ANPC 能够尝试和学习的事情之一是在给定情景 $do(a, s)$ 下,预测某些流 f_i 的值.因此,这里的目标函数是 $f_i(do(a, s))$ 的值.一般情况下, f_i 的值取决于如下因素:行动 a , 以前情景 $f_0(s), \dots, f_{n-1}(s)$ 的值.可以将学习并入效应公理中:

$$f_0(s) = v_0 \wedge \cdots \wedge f_{n-1}(s) = v_{n-1} \\ \Rightarrow f_i(do(a_i, s)) = f_i(a_i, v_0, \cdots, v_{n-1}) \quad (7)$$

式中 f_i 是学习函数, 它的参量是构成初始训练集实例的属性。

在实用世界模型中, 任一情景下, ANPC 可选择的行动可能不止一个, 对这些行动的选择涉及到对行动的强化学习, 对每个行动定义一个行动效用流, 角色通过感知来更新行动效用流, 根据行动效用流来选择行动。行动效用流可以依 Q -学习方式来更新。

所有的学习函数构成一个集合 L , L 也称为学习接口。如果记情景演算为 R , 则 ANPC 的架构可以扩展为:

$$IMP_{ANPC} = E \circ R \circ L \circ P \quad (8)$$

4 实现与应用

4.1 游戏人工智能平台

由公式(8)可知, ANPC 的架构包括感知、学习和行动三个接口以及一个扩展的情景演算模型, 而在前文里, 我们论述了如何在情景演算模型里融合感知、学习和行动这三个概念, 因此实现 ANPC 的核心便是对扩展的情景演算模型的解释以及情景演算模型如何去绑定三个接口。与实现相关的另外一个问题是命名问题。理论上, 这些命名只要符合用来实现 ANPC 的计算机语言的命名规则即可。然而, 为了达到更大的重用度, 接口里的函数和情景演算中的知识表达应该尽量按照本体来定义, 如此, 我们可以将 ANPC 的模型实现为一个用于游戏中的人工智能平台, 通过本体映射和转换平台便可以应用于不同的游戏虚拟世界。

目前关于游戏本体还没有统一的标准, 本文基于角色扮演类游戏和第一人称射击类游戏涉及的游戏概念类及其关系实现了一个简单的用于这两类游戏的人工智能平台 (ANPC-AIPlatform)。在游戏的每一帧, 对每一个 ANPC, 游戏引擎通过情景演算解释器与平台交互, 当游戏引擎第一次执行 ANPC 模型时需要调用 ANPC 的初始化例程, 初始化的工作主要是将 ANPC 情景演算中的感知、学习和行动分别与平台三个接口中的对应函数进行绑定。用户只要在平台里实现感知、学习以及行动这三个接口部分或全部函数就会产生一种 ANPC (如图 2 所示), 当然用户可以通过编辑情景演算文件对 ANPC 进行指导。

图 2 场景中, 左边列出了三个接口中已经实现的功能, 右边展示了部分功能的效果, 包括 ANPC 的自主视觉、表情生成、以及魔法学习等。很显然, 利用人工智能平台, 用户可以很方便地将自己的人工智能算法用于游戏中。

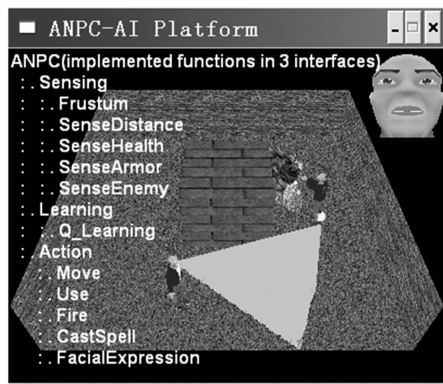


图2 用于游戏中的人工智能平台

4.2 在商业游戏中的应用

将 ANPC-AIPlatform 输出为动态链接库便可以与商业游戏引擎 Quake 2 连接。图 3 是 ANPC 与 NPC 在游戏 Quake 2 的 demo1 地图中的对比实验结果。

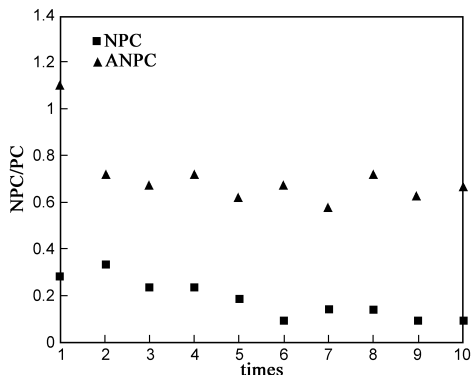


图3 不同NPC和PC对战的结果

其中横坐标表示游戏的次数, 纵坐标表示非玩家角色与玩家在死亡竞赛中的获胜次数的比值。实心矩形表示 NPC 与 PC 对战的结果, 整体数据在一个较低水平呈下降趋势, 表明随着交手的次数增多玩家对一般 NPC 越来越了解, 从而很容易就能击败它; 实心三角表示 ANPC 与 PC 对战的结果, 整体数据维持在一个较高水平, 表明 ANPC-AIPlatform 的使用, 除了易于构建自主 NPC 外, 创建的 ANPC 对玩家更具挑战性。

图 4 给出了 ANPC 行动的前提条件公理 (省略了参量 s), 描述了当 ANPC 发现真实玩家时应该采用的行动, 其中 distance、health 为语言认知流。

```

Poss( use( rocketlauncher ) )  $\Leftarrow$  enemy  $\wedge$  distance = ( far )
 $\wedge$  rocketlauncher  $\wedge$   $\neg$  ready( rocketlauncher )
Poss( use( railgun ) )  $\Leftarrow$  enemy  $\wedge$  ( distance = far )
 $\wedge$  railgun  $\wedge$   $\neg$  ready( railgun )
Poss( use( chaingun ) )  $\Leftarrow$  enemy  $\wedge$  ( distance = far )
 $\wedge$  chaingun  $\wedge$   $\neg$  ready( chaingun )
Poss( use( hyperblaster ) )  $\Leftarrow$  enemy  $\wedge$  ( distance = far )
 $\wedge$  hyperblaster  $\wedge$   $\neg$  ready( hyperblaster )
Poss( fire )  $\Leftarrow$  enemy  $\wedge$  ready_weapon
Poss( move( flee ) )  $\Leftarrow$  enemy  $\wedge$  health = low
Poss( move( seek ) )  $\Leftarrow$  enemy  $\wedge$  ( distance = close )

```

图4 采用情景演算语法给予 ANPC 的指导

5 结论

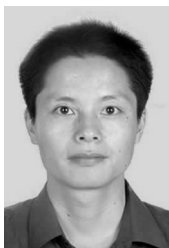
本文给出了一个基于情景演算的开放式自主非玩家角色模型. 模型将人工智能的认知建模方法和自主非玩家角色实现架构有机地融合在一起, 为设计者在更高的抽象层次上去指导角色的行动提供了便利, 基于该模型的用于游戏的人工智能平台在商业游戏引擎中的实验证实了这点, 在这个实验中仅仅给出了一个简单的游戏本体模型, 要想人工智能平台能很好地应用于更多类型更广泛的游戏, 还需要在游戏本体构建方面做大量的工作.

参考文献:

- [1] Funge J. Representing knowledge within the situation calculus using interval-valued epistemic fluents[J]. *Reliable Computing*, 1999, 5(1): 35 – 61.
- [2] Champandard A J. *AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors*[M]. Indianapolis: New Riders Publishing, 2003.
- [3] 班晓娟, 江道平, 宁淑荣, 尹怡欣. 计算机动画环境中基于认知的行为路径选择研究[J]. *电子学报*, 2009, 37(4): 758 – 763.
Ban Xiaojuan, Jiang Daoping, Ning Shurong, Yin Yixin. Research on behavior route selection from a cognitiv prospective in computer animation[J]. *Acta Electronia Sinica*, 2009, 37(4): 758 – 763. (in Chinese)
- [4] 涂序彦, 王枏, 郭燕慧. *大系统控制论(M)*. 北京: 北京邮电大学出版社, 2005. 36 – 79.

- [5] Ferrein A, Schiffer S, Lakemeyer G. A fuzzy set semantics for qualitative fluents in the situation calculus [A]. *Proc. of ICIRA 2008*[C]. Volume 5314 of *LectureNotes in Computer Science*, 2008. 498 – 509.
- [6] Mateus P, Pacheco A, Pinto J, Sernadas A, Sernadas C. Probabilistic situation calculus [J]. *Annals of Mathematics and Artificial Intelligence*, 2001, 32 (1/4): 393 – 431.

作者简介:



黄向阳 男, 1971 年出生于湖北蕲春. 首都师范大学信息工程学院讲师, 博士. 主要研究方向: 智能控制、系统仿真与虚拟现实.

E-mail: huangxiangyang@sina.com



彭 岩 女, 1967 年出生于陕西. 首都师范大学信息工程学院教授, 博士. 主要研究方向: 人工智能理论与应用、智能管理与决策.

张树东 男, 1969 年出生于河北唐山. 首都师范大学信息工程学院副研究员, 博士后. 主要研究方向: 智能信息处理、嵌入式与分布式处理系统.