UNIVERSITÉ DU QUÉBEC À MONTRÉAL

DÉVELOPPEMENT D'UN PILOTE SYNTHÉTIQUE FONDÉ SUR L'ARCHITECTURE COGNITIVE ACT-R ET OPÉRANT DANS UN SIMULATEUR DE COCKPIT

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DU DOCTORAT EN INFORMATIQUE COGNITIVE

PAR

TAMKODJOU TCHIO GUY CARLOS

NOVEMBRE 2025

REMERCIEMENTS

Au terme de ce parcours doctoral, je souhaite exprimer ma profonde gratitude à toutes les personnes et institutions qui ont contribué, de près ou de loin, à la réalisation de cette thèse.

Je tiens à adresser ma reconnaissance au professeur Roger Nkambou, mon directeur de thèse, pour sa direction éclairée et son expertise. Ses encouragements, sa patience, et sa vision scientifique ont grandement contribué à la réalisation de ce travail.

Ma co-directrice, la professeure Valéry Psyché, mérite une reconnaissance particulière. Sans son encouragement et son aide précieuse, ce doctorat n'aurait jamais vu le jour. Son soutien indéfectible, ses conseils avisés, son suivi régulier, et son expertise ont été des piliers essentiels tout au long de ce cheminement.

Ma reconnaissance va aux membres du jury, dont l'expertise et les retours avisés vont contribuer à enrichir cette thèse et à guider ma réflexion avec justesse.

Je remercie chaleureusement les professeurs et le personnel du programme du DIC qui ont significativement contribué à mon parcours académique :

- Professeur Serge Robert, dont la passion du domaine m'a permis de comprendre les fondements et nouvelles tendances en sciences cognitives.
- Professeur Roger Villemaire, qui par sa maîtrise exceptionnelle m'a guidé dans l'exploration de la représentation des connaissances tant symboliques que connexionnistes sur le plan computationnel.
- Professeur Hakim Lounis, qui m'a plongé avec passion dans la modélisation cognitive de systèmes complexes.
- Professeur Stevan Harnad, dont la simplicité m'a donné goût à la recherche grâce aux brillants invités des séminaires qu'il organise.
- Mylène Dagenais, agente de gestion aux études supérieures, pour son accompagnement administratif indispensable tout au long de ce parcours.
- Professeure Béatrice Pudelko, dont le précieux soutien a contribué à la finalisation de cette thèse.

Les membres de l'équipe du projet Pilot-AI méritent une mention spéciale pour leur précieuse collaboration et la synergie qu'ils ont apportée, constituant ainsi une source constante d'inspiration et de motivation. Je

tiens particulièrement à remercier les superviseurs Claude Frasson, Ange Adrienne Nyamen Tato et Hamdi Ben Abdessalem, ainsi que mes amis et collègues Gabrielle Joyce Tato Nana, Marc-Antoine Courtemanche, Massimo Pietracupa, et tous les autres.

À ma famille, je dédie cette thèse avec une reconnaissance infinie. À mon épouse, Mariette Gaelle, pour son amour, sa patience et son soutien constant. À mes enfants, Lilian Nathan, Carl Yohan, et Kenny Daven, qui m'ont apporté joie et motivation tout au long de ce voyage académique.

Je n'oublie pas ma mère (Kapche Marie Esther) et mon frère (Tchinda Tchio Tonton Martial) pour leur soutien inconditionnel. Un hommage particulier va à mon père (Tchio Frédéric), dont l'influence et les encouragements ont été inestimables dans ma formation académique, malgré son absence.

Un grand merci à la famille Tchuenté, à mes amis Armel Thibaut, İssa Baban Chawaï, Liliane Bawa, Charles Jules pour leur soutien et leur amitié tout au long de ce parcours.

Enfin, que toutes celles et tous ceux qui ne sont pas nommément cités dans ces pages trouvent ici toute ma reconnaissance.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

ACRONYMES

UQAM Université du Québec à Montréal.

UdeM Université de Montréal.

UQAC Université du Québec à Chicoutimi.

ETS École de Technologie Supérieure.

ULaval Université Laval.

CRIAQ Consortium de Recherche et d'Innovation en Aérospatiale au Québec.

CRSNG Conseil de Recherches en Sciences Naturelles et en Génie du Canada.

CAE Canadian Aviation Electronics.

BMU Beam Me Up.

IRM Imagerie par résonance magnétique.

IRMf Imagerie par résonance magnétique fonctionnelle.

EEG Électroencéphalogramme.

SWRL Semantic Web Rule Language.

DIC Doctorat en Informatique Cognitive.

API Application Programming Interface.

IA Intelligence Artificielle.

AI Artificial Intelligence.

Pilot-AI AI Augmented Pilot.

C-Pilot Cognitive Pilot.

RÉSUMÉ

Le pilotage d'un avion est une tâche complexe qui nécessite un accompagnement des pilotes par des systèmes artificiels fiables. Cependant, la résolution des situations critiques qui apparaissent dans un avion dépend toujours des pilotes, qui doivent intervenir en assumant le contrôle manuel de l'appareil si cela s'avère nécessaire. De plus, il est fréquent que les pilotes subissent des baisses d'attention entraînant une diminution de leur performance dans le cockpit. Face à cette dégradation de la performance des pilotes, il devient important qu'ils bénéficient d'une assistance cognitive. La qualité de l'assistance qui peut être fournie dépend donc de la tâche, de l'environnement de la tâche et des processus cognitifs du pilote.

Pour représenter formellement les connaissances relatives aux tâches de pilotage et à leur environnement, les ontologies offrent une solution pertinente. Le modèle de référence ontologique, issu de travaux récents en modélisation des connaissances dans le domaine aéronautique, permet de formaliser les connaissances procédurales complexes associées aux procédures de pilotage. Ce modèle, structuré en deux composantes complémentaires (une ontologie du domaine et une ontologie des tâches), permet la manipulation automatique des connaissances en détaillant, pour chaque tâche, les informations contextuelles nécessaires à son exécution.

Pour les besoins de notre recherche, nous avons enrichi le modèle de référence ontologique en y intégrant l'ontologie Air Traffic Management (ATM) développée par la NASA (National Aeronautics and Space Administration), afin de l'adapter aux exigences de nos simulations. L'ontologie de référence ainsi obtenue formalise les connaissances expertes du domaine aéronautique telles qu'elles sont définies dans les standards, recommandations et bonnes pratiques du secteur.

Les architectures cognitives sont des cadres théoriques et computationnels utilisés pour modéliser et simuler les processus cognitifs humains. En nous appuyant sur les propriétés de l'architecture cognitive recherchée, sur les critères de Newell et sur les désiderata de Sun permettant de définir diverses capacités, propriétés et critères d'évaluation des architectures cognitives, nous avons choisi ACT-R (Adaptive Control of Thought – Rational), qui est une architecture cognitive complète et scientifiquement fondée, ayant produit des modèles qui ont permis de reproduire le comportement d'un agent dans un contexte de pilotage.

Notre étude a conduit au développement d'un pilote synthétique fondé sur l'architecture cognitive ACT-R, exploitant l'ontologie de référence des procédures de pilotage pour exécuter des tâches conformément aux recommandations des experts de l'aviation, en situations normales comme anormales. Le modèle intègre les principales fonctions cognitives humaines, notamment la perception, la mémoire, l'apprentissage et l'action, ainsi que leurs limites. En exploitant cette connaissance experte fournie par l'ontologie et en utilisant les mécanismes d'apprentissage par renforcement et statistique propres à ACT-R, notre système offre une assistance cognitive aux pilotes humains, contribuant ainsi à l'amélioration de leur formation et à l'optimisation de leur processus de prise de décision.

Mots clés : Architecture cognitive, ACT-R, Pyactr, ontologie, ontologie du domaine, ontologie des tâches, ontologie de référence, SWRL, graphe, agent cognitif.

INTRODUCTION

0.1 Introduction et mise en contexte

De nos jours, les raisons qui poussent les hommes à voyager sont nombreuses et diverses, et les moyens de transport abondent. Cependant, avec la multiplication des compagnies de transport aériennes à bas prix, le secteur du transport aérien attire de plus en plus de voyageurs. Dans l'aviation, on utilise des méthodes et des techniques coûteuses pour éviter des défaillances. C'est pourquoi les systèmes aéronautiques suivent des normes et des certifications qui garantissent leur fiabilité, leur sécurité, leur tolérance aux fautes (RTCA, 2012), ce qui fait de l'avion le moyen de transport le plus sûr au monde (International Air Transport Association, 2024). Pour garder ce statut, l'OACI, une institution des Nations Unies chargée de mettre en place des normes et des procédures internationales permettant de réglementer le secteur aéronautique, essaie en permanence d'améliorer la sécurité aérienne (OACI, 2016).

Cependant, malgré toutes les précautions prises dans l'aviation, des accidents continuent de survenir, leurs causes étant multiples. Ainsi, la plupart des accidents d'aviation se produisent à une hauteur relativement faible, soit avant, pendant ou après le décollage ou l'atterrissage. Un nombre plus faible d'accidents se produit en plein vol. Les chiffres indiquent que la phase de décollage, qui représente 2% de la durée du vol, enregistre 30% des accidents mortels. La phase d'atterrissage, quant à elle, représentant 4% de la durée du vol, enregistre 25% des accidents mortels. 20% des accidents sont causés par des problèmes techniques, 12% ont pour cause la météo, 8% sont l'œuvre de sabotages, 53% sont causés par des erreurs de pilotage, 6% par d'autres erreurs humaines extérieures au pilotage, comme l'entretien de l'aéronef, et 1% des accidents sont liés à d'autres causes (Clifford, 2022).

Face à ce constat, des acteurs majeurs de l'industrie aéronautique tels que BMU, CAE et Bombardier, soutenus par des organismes de recherche comme le CRIAQ et le CRSNG, ont entrepris d'améliorer la sécurité dans l'aviation civile. Cette initiative a donné naissance au projet Pilot-AI, mis en œuvre par l'UQAM et l'UdeM, deux universités montréalaises. Pilot-AI se présente comme un projet alliant des dimensions informatiques et cognitives. Notre recherche s'inscrit dans cette perspective, avec pour objectif de développer un agent cognitif capable d'assister le pilote dans l'analyse et l'exécution des tâches de pilotage, en conformité avec les procédures en vigueur, afin d'améliorer sa formation et sa prise de décision.

0.2 Objectifs de la thèse

De la section précédente, il découle que la majorité des accidents sont l'œuvre d'erreurs humaines de pilotage et se produisent généralement pendant les phases de décollage et d'atterrissage. Selon Jean-Pierre Otelli (2022), l'homme est dans la majorité des cas à l'origine des accidents, faisant ainsi du facteur humain le maillon faible de la sécurité aérienne. Selon lui, les erreurs de pilotage sont nombreuses et variées, et les acteurs mis en cause se retrouvent aussi bien dans la catégorie des bons pilotes que dans celle des pilotes moins expérimentés. Parmi les erreurs de pilotage identifiées, on peut citer, sans être exhaustif : le manque de formation, une mise à jour approximative, des négligences routinières, le non-respect des procédures, un ego surdimensionné, une autosatisfaction dangereuse, un machisme insupportable, la corruption, l'emprise de l'alcool, le stress, la distraction, la maladresse, la fatigue, la maladie et parfois même des problèmes psychiatriques.

Un accident d'avion résulte généralement d'une série d'erreurs de pilotage qui se succèdent, compromettant fatalement l'aéronef et causant la mort des passagers et de l'équipage. Dans ces situations dramatiques, l'équipage perd graduellement la maîtrise de l'appareil jusqu'à atteindre un seuil critique au-delà duquel la récupération devient impossible.

Pendant un vol, une manœuvre incorrectement exécutée par un pilote peut entraîner une déviation de la trajectoire de l'appareil. Il n'est pas rare que plusieurs écarts successifs soient mal corrigés par l'équipage, aggravant progressivement la situation. Dans certains cas favorables, l'équipage parvient à reprendre le contrôle de l'avion avant d'atteindre un point critique. Malheureusement, dans d'autres circonstances, l'issue est catastrophique.

Face à ce constat, il devient essentiel de réduire le nombre d'accidents imputables aux erreurs de pilotage pendant les phases de décollage et d'atterrissage, ce qui permettrait d'accroître la sécurité des voyages en avion et de maintenir l'avion en tête du classement des moyens de transport les plus sûrs.

Plusieurs questions de recherche émergent de cette problématique et nécessitent des réponses appropriées :

— un pilote synthétique qui simule la cognition humaine et qui s'appuie sur des connaissances expertes des procédures de pilotage d'un avion, peut-il assister efficacement un pilote humain dans l'exécution de ses tâches, améliorant ainsi sa formation et sa capacité de décision ?

— comment construire un tel pilote synthétique ?

Pour répondre à ces questions de recherche, nous formulons les hypothèses de recherche suivantes :

— L'architecture cognitive ACT-R permet de modéliser le comportement d'un pilote humain en intégrant diverses capacités cognitives, telles que la perception, l'attention, la sélection des actions, la mémoire, l'apprentissage et le raisonnement.
— Une ontologie de référence des procédures de pilotage des avions peut modéliser les connaissances procédurales complexes liées aux tâches de pilotage et structurer les informations essentielles à leur exécution en fonction de l'environnement.
— L'intégration d'une ontologie de référence des procédures de pilotage des avions dans un agent cognitif fondé sur ACT-R peut permettre d'assister efficacement les pilotes humains, tout en renforçant leur formation et leur capacité de prise de décision.

La présente thèse vise à développer un pilote synthétique fondé sur une architecture cognitive ACT-R et opérant dans un cockpit.

Il existe trois approches permettant de construire un pilote synthétique ou virtuel :

— La première consiste à modéliser le pilote synthétique comme un agent basé sur l'apprentissage par renforcement ;
— La seconde le conçoit comme un pilote moyen utilisant un modèle d'actions apprises ;
— La troisième, que nous adopterons, repose sur un agent cognitif fondé sur une architecture cognitive donnée. Notre approche se fondera sur l'architecture cognitive ACT-R.

Ce pilote synthétique s'appuiera sur une ontologie de référence structurant les connaissances expertes relatives aux procédures de pilotage d'un avion. Il intégrera les mécanismes d'apprentissage propres à l'architecture ACT-R, tels que l'apprentissage par renforcement et l'apprentissage statistique, afin de reproduire les processus cognitifs humains. Une telle approche vise à renforcer la formation des pilotes et à optimiser leur prise de décision, aussi bien dans des décollages en procédure normale que dans des situations anormales.

0.3 Structure de la thèse

Nous avons structuré cette thèse en sept chapitres. Le chapitre 0, consacré à l'introduction générale, nous a permis de présenter le contexte général dans lequel se situe cette recherche tout en précisant les objectifs de la thèse ainsi que la problématique, tant sur le plan cognitif qu'informatique.

Le premier chapitre est consacré à la présentation du cadre théorique fondamental pour le développement du pilote synthétique. Nous y proposons une analyse détaillée des architectures cognitives candidates pour la modélisation de notre pilote synthétique, notamment ACT-R, SOAR, CLARION, SPAUN et LEABRA, en mettant en lumière leurs approches distinctives pour reproduire les processus mentaux complexes. Nous exposons également les critères ayant guidé la sélection d'ACT-R comme architecture cognitive retenue.

Le deuxième chapitre détaille les principes de fonctionnement du pilote automatique et du pilote synthétique, en mettant en évidence les différences fondamentales entre ces deux approches. Nous justifions également le choix du modèle de référence ontologique pour structurer les connaissances nécessaires aux tâches de pilotage.

Dans le chapitre 3, nous montrons comment intégrer une ontologie de référence des procédures de pilotage dans un agent cognitif ACT-R, afin de simuler une tâche spécifique du décollage d'un avion de manière fidèle au comportement d'un pilote humain expert. Nous décrivons en détail le processus de modélisation cognitive mis en œuvre, en insistant sur les mécanismes de perception, de prise de décision et d'action activés par le pilote synthétique à partir de ses connaissances déclaratives et procédurales.

Le chapitre 4 étend les capacités de l'agent cognitif présenté au chapitre précédent en y intégrant un modèle de coaching cognitif conçu pour assister les pilotes en temps réel lors de l'exécution de la procédure normale et complète de décollage. Ce pilote synthétique est capable de détecter les écarts par rapport aux procédures standard et de formuler des recommandations adaptées afin de corriger ces déviations de manière efficace.

Le chapitre 5 étend encore les capacités de l'agent cognitif en y intégrant la gestion des situations urgentes et anormales pouvant survenir au décollage, ainsi qu'un modèle de coaching cognitif conçu pour assister les pilotes en temps réel. Cette extension vise à accompagner les pilotes tant dans leurs processus d'apprentissage que dans leur prise de décision, que ce soit lors de décollages normaux ou en présence de situations

anormales. Le pilote synthétique est capable de détecter les écarts par rapport aux procédures standard et de formuler des recommandations adaptées pour corriger efficacement ces déviations. Les situations critiques prises en charge incluent notamment une panne moteur après la vitesse V1, un décollage interrompu après V1, un cisaillement du vent lors du décollage, une alerte du système de trafic et d'évitement de collision (TCAS), une double panne moteur avec carburant restant, ainsi qu'une récupération après un décrochage.

Le chapitre 6 est consacré à la discussion des résultats obtenus. Il présente une synthèse des différentes études réalisées dans le cadre de cette thèse. Il permet de prendre du recul sur l'ensemble du travail effectué, d'en tirer des conclusions globales et de souligner la portée et l'impact de cette recherche dans les domaines des sciences cognitives et de l'informatique. Enfin, il explore les modalités par lesquelles le pilote synthétique pourrait évoluer vers un véritable pilote virtuel.

Enfin, la conclusion générale vient clôturer cette recherche. Elle résume les principaux apports de notre étude et propose une analyse critique des contributions. Ces contributions ouvriront des perspectives pour de futures recherches.

CHAPITRE 1
CADRE THÉORIQUE

1.1 Introduction

Dans ce chapitre, nous proposons un aperçu des recherches menées dans le domaine des architectures cognitives les plus populaires, notamment ACT-R, SOAR, CLARION, SPAUN et LEABRA. Nous avons jugé que ces architectures pourraient être des candidates sur lesquelles sera fondé notre pilote synthétique.

Cette exploration vise à comprendre les approches développées pour modéliser la cognition humaine et identifier les mécanismes qui permettent de reproduire les processus mentaux complexes. Ces architectures cognitives représentent des tentatives de créer des systèmes capables de penser, percevoir, apprendre et raisonner de manière similaire à l'intelligence humaine.

Notre revue détaillée vise à mettre en lumière les caractéristiques distinctives de chaque architecture, leurs forces et limites, ainsi que leurs contributions potentielles à la compréhension des mécanismes cognitifs. En examinant ces modèles, nous cherchons à identifier les approches les plus prometteuses pour développer un pilote synthétique capable de reproduire les processus cognitifs complexes d'un pilote humain.

Nous débuterons par une définition des architectures cognitives, en expliquant leur rôle dans la modélisation des mécanismes cognitifs humains et leur importance dans la recherche en IA. Ensuite, nous mettrons en avant les enjeux majeurs et les critères d'évaluation de ces architectures, en insistant sur leurs capacités cognitives fondamentales et leurs applications pratiques.

Nous étudierons également les trois principaux paradigmes des architectures cognitives, à savoir le symbolique, le connexionniste et l'hybride, ainsi que leurs implications respectives dans le domaine de l'IA.

Enfin, nous décrirons en détail ces architectures cognitives de référence qui se distinguent par leur structure et leur champ d'application.

1.2 Les architectures cognitives

1.2.1 Définitions

Newell (1990), dans son livre *Unified Theories of Cognition*, décrit les architectures cognitives en ces termes :

> « … *Un système unique (esprit) produit tous les aspects du comportement. C'est un seul esprit qui s'occupe de tous ceux-ci. Même si l'esprit est composé de parties, modules, composantes, ou de quoi que ce soit, ceux-ci se tissent ensemble pour produire le comportement … Si une théorie ne couvre qu'une partie ou un composant, elle courtise le danger dès le départ. Il va sans dire qu'il y a des dissociations, des interdépendances, des impénétrabilités et des modularités … Mais elles n'évacuent pas la nécessité d'une théorie qui offre la vue d'ensemble, et explique le rôle des parties et pourquoi elles existent.* »

Les théories computationnelles, appelées *théories unifiées de la cognition*, définissent une architecture cognitive comme un système unique dont l'ensemble des mécanismes tente d'expliquer la diversité des comportements cognitifs humains (résolution de problème, prise de décisions, routines, action, mémorisation, apprentissage, activité motrice, langage, motivation, émotion, imagination, rêve, etc.), offrant ainsi une compréhension unifiée de ces phénomènes.

Pour Kotseruba et Tsotsos (2020), une architecture cognitive est définie comme la combinaison d'outils informatiques conçus pour générer une capacité de perception, de réflexion et de prise de décision similaire à celle d'un être humain.

L'objectif des Architectures Cognitives est de modéliser l'esprit humain avec sa variété de capacités cognitives, notamment la perception, les mécanismes d'attention, la sélection d'actions, la mémoire, l'apprentissage, le raisonnement et le méta-raisonnement. À terme, cela permettra de concevoir une IA à l'échelle humaine et d'explorer les quatre dimensions fondamentales de l'IA, telles que définies dans Russell et Norvig (2020) :

- Penser comme un humain ;
- Penser rationnellement ;
- Agir comme un humain ;
- Agir rationnellement.

Ces dimensions sont illustrées dans la Figure 1.1, adaptée de Université TÉLUQ (2019).

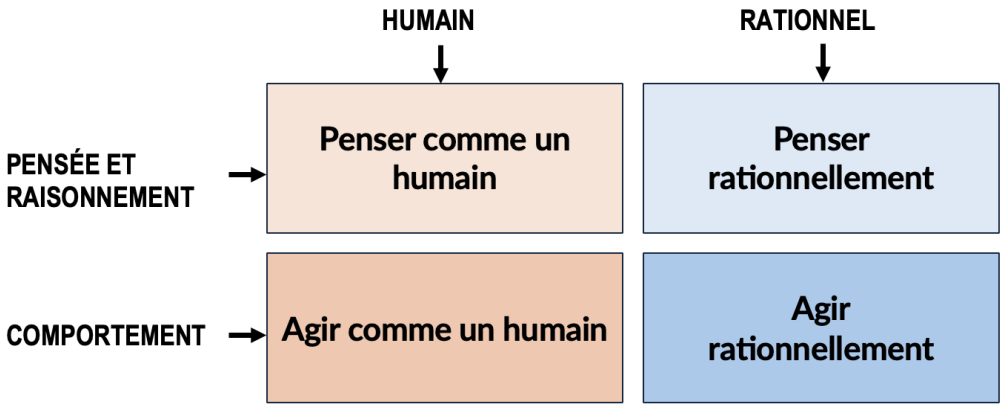


Figure 1.1 – Les quatre dimensions fondamentales de l'IA.

Les quatre approches de l'IA présentées à la Figure 1.1 permettent d'analyser l'IA sous quatre perspectives distinctes. La première ligne s'intéresse aux processus de pensée et de raisonnement, tandis que la seconde met l'accent sur le comportement. Par ailleurs, la colonne de gauche évalue la réussite en fonction des performances humaines, alors que celle de droite mesure la rationalité (Université TÉLUQ, 2019).

Dans ce cadre, plusieurs auteurs ont défini l'IA selon ces différentes perspectives, comme l'illustrent les citations suivantes :

- Penser comme un humain : « La tentative nouvelle et passionnante d'amener les ordinateurs à penser … [d'en faire] des machines dotées d'un esprit au sens le plus littéral. » (Haugeland, 1985) ou « [L'automatisation d'] activités que nous associons à la pensée humaine, des activités telles que la prise de décision, la résolution de problèmes, l'apprentissage […] » (Bellman, 1978). Exemple : Architectures cognitives comme ACT-R, SOAR, etc.
- Agir comme un humain : « L'art de créer des machines capables de prendre en charge des fonctions exigeant de l'intelligence quand elles sont réalisées par des gens. » (Kurzweil, 1990) ou « L'étude des moyens à mettre en œuvre pour faire en sorte que des ordinateurs accomplissent des choses pour lesquelles il est préférable de recourir à des personnes pour le moment. » (Rich et Knight, 1991). Exemple : Test de Turing, Chatbots comme ChatGPT, etc.
- Penser rationnellement : « L'étude des moyens informatiques qui rendent possibles la perception, le raisonnement et l'action. » (Winston, 1992) ou « L'étude des facultés mentales grâce à des modèles

informatiques. » (Charniak et McDermott, 1985). Exemple : Logique formelle.

— Agir rationnellement : « L'intelligence artificielle (computational intelligence) est l'étude de la conception d'agents intelligents. » (Poole *et al.*, 1998) ou « L'IA [...] étudie le comportement intelligent dans des artefacts. » (Nilsson, 1998). Exemple : Voitures autonomes.

1.2.2 Enjeux, critères d'évaluation, capacités cognitives fondamentales et applications pratiques des architectures cognitives

1.2.2.1 Enjeux

La question des architectures cognitives occupe une place importante dans les recherches en sciences cognitives, tant en psychologie cognitive, en IA, qu'en philosophie de l'esprit.

Dans la littérature, la recherche sur les architectures cognitives a pour objectif de développer une théorie capable de rendre compte de l'ensemble des mécanismes cognitifs dont l'interaction est responsable du comportement global d'un agent.

Le but ultime de la recherche sur les architectures cognitives est de modéliser l'esprit humain, ce qui permettra à terme de construire une IA à l'échelle humaine.

Dans Lieto *et al*. (2018), les architectures cognitives permettent de :

— Raviver le rêve de l'IA forte ;

— Tester les théories cognitives en les implémentant, sans les contraintes éthiques liées à l'expérimentation sur des sujets humains ;

— Rapprocher la programmation des découvertes en neurosciences ;

— Identifier les limites des approches symbolistes et naturalistes et proposer des alternatives pour dépasser cette opposition ;

— Améliorer les performances de l'IA en affinant notre compréhension des composantes de la cognition, de leur fonctionnement et de leurs interactions.

1.2.2.2 Critères d'évaluation des architectures cognitives

Les critères de Newell (Anderson et Lebiere, 2003; Newell, 1980, 1992) et les desiderata de Sun (Sun, 2004), repris dans Kotseruba et Tsotsos (2020), définissent diverses capacités, propriétés et critères d'évalua-

tion des architectures cognitives, qui comprennent la reconnaissance, la prise de décision, la perception, la prédiction, la planification, l'action, la communication, l'apprentissage, la fixation d'objectifs, l'adaptabilité, la généralité, l'autonomie, la résolution de problèmes, le fonctionnement en temps réel, le méta-apprentissage, etc. (Asselman *et al.*, 2015; Langley *et al.*, 2009; Thórisson et Helgasson, 2012; Vernon *et al.*, 2007).

Ces critères d'évaluation sont résumés et décrits dans le tableau ci-dessous. Ils peuvent être regroupés en cinq catégories (Anderson et Lebiere, 2003; Eliasmith, 2013; Langley *et al.*, 2009; Newell, 1990; Sun, 2004) :

- Le comportement ;
- La cognition ;
- Les réalismes ;
- Les propriétés représentationnelles ;
- La validité scientifique.

Catégorie	Description
1. Comportement	**1.1 Rationalité** : Elle s'évalue en étudiant la relation entre les actions du système en rapport aux connaissances et buts qu'il possède.
	1.2 Généralité, versatilité et généricité : - Généralité : Elle permet au système de s'adapter à une variété de tâches et d'environnements. - Versatilité : Elle s'évalue par la mesure des efforts requis de la part du développeur pour adapter le système à chaque nouvelle tâche et nouvel environnement. - Généricité : Elle désigne l'aptitude de l'architecture à réaliser différentes tâches sur le simple envoi d'une commande.
	1.3 Réactivité : Elle désigne l'aptitude à réagir à l'inattendu.
	1.4 Robustesse : Elle désigne l'aptitude à maintenir un but dans un environnement non prévisible, avec des données imprécises ou bruitées. C'est aussi l'aptitude de l'architecture à maintenir une bonne performance au cours de l'enchainement de plusieurs tâches.
	1.5 Séquentialité et Routine : Elle décrit la capacité à décomposer les activités en étapes séquentielles et à les exécuter de manière routinière.

Catégorie	Description
	1.6 Adaptabilité : Elle désigne l'aptitude d'un système à tenir compte dans ses actions futures de ses expériences passées.
	1.7 Efficacité et mise à l'échelle : Elle exige que le système fonctionne à des vitesses comparables à celles des humains, tout en optimisant l'utilisation du temps et des ressources.
	1.8 Autonomie et opération étendue : Elle permet au système de générer ses propres objectifs pour fonctionner de manière autonome sur de longues périodes.
2. Cognition	**2.1 Langage naturel** : Le système se doit d'inclure des hypothèses quant au langage. Cet aspect repose sur la manipulation des symboles.
	2.2 Dichotomie des processus implicites et explicites : Les processus et connaissances implicites sont difficilement accessibles alors que les processus et connaissances explicites sont à l'inverse facilement accessibles.
	2.3 Conscience : La manifestation d'une conscience ou sens de soi.
	2.4 Généralisation syntaxique : La capacité à généraliser sur la base de la structure syntaxique des représentations.
	2.5 Apprentissage : Le système doit apprendre.
	2.6 Modularité : Les facultés cognitives doivent être représentées au sein d'une architecture cognitive dans des modules distincts.
	2.7 Mémoire : Elle doit rendre compte des différentes fonctions associées aux mémoires (mémoire de travail, mémoire à long terme).
3. Réalismes	**3.1 Réalisme cognitif** : Il faut tenir compte des caractéristiques connues du comportement humain et de ses différentes fonctions cognitives.
	3.2 Réalisme biologique : On doit pouvoir associer les processus qui se trouvent dans l'architecture à des régions cérébrales.
	3.3 Réalisme évolutif : Il faut tenir compte dans la conception de l'architecture cognitive des théories évolutives.
	3.4 Réalisme écologique : L'agent cognitif doit être situé dans son environnement et avoir les fonctions suffisantes pour réagir et s'adapter à son environnement.

Catégorie	Description
4. Propriétés représentationnelles	**4.1 Bases de connaissances vastes** : Les connaissances doivent reproduire la complexité du monde réel.
	4.2 Variétés de types de connaissances : Les connaissances à manipuler doivent être variées.
	4.3 Systématicité : Le caractère systématique de nos pensées, se manifestant par la connexion nécessaire entre différentes pensées, doit être retranscrit quel que soit le type de représentation choisi.
	4.4 Compositionalité : L'architecture doit être capable comme l'humain de combiner ses connaissances.
	4.5 Liage massif : L'architecture cognitive doit avoir la capacité de générer des structures complexes dans une durée de temps appropriée.
	4.6 Productivité : L'architecture cognitive doit avoir la capacité de générer une variété infinie de phrases grammaticales avec un ensemble fini de mots et une grammaire finie.
5. Validité scientifique	**5.1 Triangulation** : La théorie unifiée de la cognition sur laquelle une architecture cognitive est basée doit être complète et ainsi présenter clairement sa relation à toutes les disciplines qui visent à comprendre l'organisation fonctionnelle et biologique du cerveau.
	5.2 Simplicité : La théorie doit être décrite de manière brève.

Table 1.1 – Critères d'évaluation des architectures cognitives

En plus des critères d'évaluation définis ci-dessus, les architectures cognitives sont également classées selon le type de représentation et de traitement de l'information qu'elles utilisent. Trois grands paradigmes sont actuellement reconnus : symbolique, connexionniste, et hybride (Kotseruba et Tsotsos, 2020). Depuis plus de trois décennies, la communauté scientifique débat activement pour déterminer si l'un de ces modèles représente avec précision les mécanismes cognitifs humains, une question qui demeure non résolue à ce jour (Kelley, 2003; Oltramari et Sun, 2025).

1.2.2.2.1 Symbolique

Le cognitivisme est une doctrine en IA qui considère la cognition comme un processus de manipulation de symboles pour représenter, traiter et transformer l'information (Bastien, 2017). Il postule que l'esprit humain fonctionne comme un système de traitement de l'information, où les connaissances sont stockées sous forme de représentations mentales et manipulées par des règles logiques. Il domine les sciences cognitives du milieu des années 1950 aux années 1980 et compare l'esprit à un ordinateur (McGill, 2016). Il développe les idées de :

- Thomas Hobbes pour qui penser, c'est manipuler des symboles et raisonner, c'est calculer (Hobbes, 1651).
- George Boole qui considère que les lois de la pensée sont des règles formelles pour la manipulation des propositions (Boole, 1854).
- Herbert Simon et Avron Barr selon qui l'IA fournit le modèle informatique de la cognition (Simon, 1996; Barr et Feigenbaum, 2014).

En 1975, le philosophe Jerry Fodor publie *The Language of Thought*, où il propose un modèle de la pensée inspiré de l'analogie avec le fonctionnement de l'ordinateur (Fodor, 1975). Selon lui, la pensée est au cerveau ce que le logiciel (software) est à la machine (hardware) (McGill, 2016). Cette approche influence fortement le développement de l'IA symbolique.

L'IA symbolique repose sur la manipulation de symboles et de règles logiques pour simuler le raisonnement humain. Elle utilise des représentations symboliques lisibles par l'homme pour modéliser les connaissances et les processus cognitifs (Garnelo et Shanahan, 2019). Cette approche est directement inspirée du cognitivisme et de la théorie du langage de la pensée de Jerry Fodor.

Une approche courante dans la conception des systèmes cognitifs consiste à modéliser un mécanisme de décision basé sur une représentation symbolique. En utilisant des symboles et des prédicats, il est possible de tirer des conclusions à condition qu'un ensemble de connaissances soit explicitement défini. Toutefois, dans ce cadre, tout événement non spécifié est automatiquement considéré comme faux, ce qui limite l'adaptabilité du système (Kotseruba et Tsotsos, 2020).

1.2.2.2.2 Connexionniste

Le connexionnisme est un domaine de recherche qui se concentre sur les systèmes composés d'un grand nombre d'unités de calcul interconnectées. Ce paradigme s'inspire du fonctionnement du cerveau humain et cherche à reproduire des processus cognitifs à l'aide de réseaux de neurones artificiels (Heudin, 2017).

Un réseau de neurones artificiels est un ensemble de neurones artificiels interconnectés, inspirés par les cellules nerveuses du cerveau. Ces réseaux sont au cœur de l'IA connexionniste, un domaine qui s'efforce de modéliser l'intelligence humaine en imitant la manière dont le cerveau traite l'information (Touzet, 1992).

Les travaux de plusieurs chercheurs pionniers ont été fondamentaux dans le développement du connexionnisme :

- Warren McCulloch et Walter Pitts, deux neurologues, ont proposé de simuler le fonctionnement interne du cerveau humain dans une machine. Ils ont inventé le neurone formel, qui est le premier modèle mathématique du neurone (McCulloch et Pitts, 1943).
- Donald Hebb, dans son ouvrage *The Organization of Behavior*, a formulé la règle d'apprentissage de Hebb, selon laquelle la force des connexions neuronales augmente lorsque deux neurones sont activés simultanément. Cette idée est résumée par la phrase *« neurons that fire together wire together »* (Hebb, 1949).
- Frank Rosenblatt a développé le Perceptron, un modèle de réseau neuronal capable d'apprentissage supervisé (Rosenblatt, 1958).

L'IA connexionniste se distingue de l'IA symbolique par son approche : alors que l'IA symbolique repose sur la manipulation de symboles et de règles explicites, l'IA connexionniste repose sur des réseaux de neurones interconnectés capables d'apprendre à partir de données grâce à des algorithmes d'apprentissage. Cette approche repose sur un vaste ensemble de composants interconnectés, à l'image d'un réseau neuronal. L'information se propage à travers ce réseau, conduisant l'architecture cognitive à prendre des décisions ou à exécuter des actions. Bien que cette approche soit plus adaptée aux environnements dynamiques et changeants, elle requiert un entraînement intensif. De plus, une fois formé, le système risque d'altérer son apprentissage initial en intégrant de nouvelles données (Kotseruba et Tsotsos, 2020).

1.2.2.2.3 Hybride

Les architectures cognitives hybrides représentent un domaine clé dans la recherche en IA, combinant les forces de l'approche symbolique et de l'approche connexionniste pour surmonter les limitations propres à chaque paradigme pris isolément (Kotseruba et Tsotsos, 2020).

L'approche symbolique repose sur la manipulation explicite de symboles et de règles logiques pour simuler des processus cognitifs. Elle est particulièrement efficace pour gérer des connaissances structurées, des raisonnements complexes et des tâches de planification. Cependant, elle se heurte à des difficultés lorsqu'il s'agit de traiter des informations sensorielles complexes, de faire face à l'incertitude ou d'adapter les comportements en temps réel dans des environnements dynamiques.

À l'inverse, l'approche connexionniste, qui s'appuie sur des réseaux neuronaux et des systèmes inspirés du cerveau humain, excelle dans l'apprentissage à partir de données non structurées et la gestion de situations incertaines ou imprévues. Elle permet aux systèmes d'apprendre de manière adaptative et de traiter des informations sensorielles complexes en temps réel, mais elle manque souvent de transparence, ce qui rend difficile l'interprétation des décisions prises par ces systèmes.

Les systèmes hybrides tirent parti des points forts de ces deux approches en intégrant des composants symboliques et connexionnistes dans une architecture cohérente. Dans les architectures mobiles comme les robots ou les agents intelligents, ces systèmes adoptent généralement une structure principalement symbolique pour gérer des tâches de haut niveau, comme la planification, la prise de décision logique et la représentation explicite des connaissances. Parallèlement, des éléments connexionnistes, comme des réseaux de neurones, sont intégrés pour gérer le traitement des données sensorielles en temps réel, l'apprentissage adaptatif et la gestion de l'incertitude.

Cette combinaison permet non seulement de pallier les défauts de chaque approche, mais aussi d'offrir une plus grande flexibilité et efficacité dans des environnements complexes et en constante évolution. Les architectures hybrides sont particulièrement adaptées aux robots autonomes, où une prise de décision rapide et une adaptation continue sont essentielles. Elles permettent aux robots d'apprendre de leurs interactions avec l'environnement tout en maintenant une base de connaissances stable et structurée pour des tâches cognitives complexes (Kotseruba et Tsotsos, 2020).

1.2.2.3 Capacités cognitives fondamentales

Le but des architectures cognitives est de modéliser l'esprit humain. Les capacités cognitives fondamentales à mettre en œuvre sont : la perception, les mécanismes d'attention, la sélection d'actions, la mémoire, l'apprentissage, le raisonnement et le méta-raisonnement (Kotseruba et Tsotsos, 2020).

1.2.2.3.1 La perception

La perception joue un rôle essentiel dans la cognition humaine. Elle constitue un processus qui transforme l'entrée brute en une représentation interne du système, facilitant ainsi l'exécution des tâches cognitives. On distingue plusieurs modalités sensorielles, dont les cinq plus courantes sont : la vision, l'audition, l'odorat, le toucher et le goût (Kotseruba et Tsotsos, 2020).

D'autres sens humains existent également, tels que :

- La proprioception ou sensibilité profonde : perception, consciente ou non, de la position des différentes parties du corps (Héroux *et al.*, 2022) ;
- La thermoception : sensation et perception de la température (Vabba *et al.*, 2023) ;
- La nociception : perception de la douleur, jouant un rôle défensif et d'alerte (Armstrong et Herr, 2023) ;
- La perception du temps qui est une construction mentale qui varie selon notre attention, notre état émotionnel, et le contexte dans lequel on vit une expérience (Droit-Volet, 2025).
- etc.

Les architectures cognitives intègrent certaines modalités sensorielles à l'aide de divers dispositifs, tels que la saisie symbolique, les capteurs, les lasers, les boussoles, les caméras et les gyroscopes (González-Santamarta *et al.*, 2025).

1.2.2.3.2 Les mécanismes d'attention

L'attention joue un rôle essentiel dans la cognition humaine, car elle sert de médiateur dans la sélection des informations pertinentes et filtre les informations non pertinentes issues des données sensorielles entrantes. Elle regroupe un ensemble de mécanismes influençant à la fois les processus perceptifs et cognitifs (Tsotsos et Kruijne, 2014).

Parmi les différentes formes d'attention, l'attention visuelle est la plus étudiée.

Les principaux mécanismes de l'attention sont (Tsotsos, 2011) :

- La sélection : identification et focalisation sur les informations pertinentes ;
- La restriction : limitation du traitement aux éléments essentiels ;
- La suppression : inhibition des informations non pertinentes.

1.2.2.3.3 La sélection d'actions

La sélection des actions détermine à chaque instant ce qu'il convient de faire ensuite. Elle se divise en deux parties (Ozturk, 2009) :

- La partie « quoi », qui implique la prise de décision ;
- La partie « comment », qui concerne le contrôle des moteurs.

On distingue deux grandes approches de sélection d'actions (Kotseruba et Tsotsos, 2020) :

- La planification : Elle fait référence aux algorithmes traditionnels d'IA permettant de déterminer une séquence d'étapes pour atteindre un certain objectif ou résoudre un problème ;
- La sélection dynamique d'actions : Elle permet de choisir une meilleure action parmi les alternatives en fonction des connaissances disponibles à ce moment. Les critères de sélection des actions dépendent de facteurs tels que la pertinence, les motivations, les humeurs, les pulsions, les émotions, etc.

Dans les architectures cognitives, les émotions artificielles sont généralement modélisées comme des états transitoires associés à des sentiments tels que la colère, la peur, la joie, etc. (Hudlicka, 2004). Elles influencent les capacités cognitives et jouent un rôle dans le processus de sélection des actions.

1.2.2.3.4 La mémoire

Il existe chez l'homme des mémoires en interaction : la mémoire sensorielle, la mémoire à court terme et la mémoire à long terme. La communication entre ces mémoires est due au fait que les informations, avant d'être présentes dans la mémoire à court terme, sont captées par la mémoire sensorielle puis stockées dans la mémoire à long terme. Elles peuvent également retourner dans la mémoire à court terme et ainsi de suite (Atkinson et Shifrin, 1968).

Dans la mémoire à court terme, les informations durent environ 30 secondes. C'est la mémoire de travail. Le nombre moyen d'éléments que l'on peut traiter simultanément sur une courte durée dans celle-ci est égal à sept, plus ou moins deux (Miller, 1956).

La mémoire à long terme, quant à elle, stocke des informations dont la durée est variable. Le stockage peut durer des années, voire toute la vie. C'est notre disque dur. C'est la mémoire des procédures. Les contenus peuvent être épisodiques ou sémantiques (Nicolas, 2002).

La mémoire est un composant essentiel de la cognition. Elle constitue une partie fondamentale d'une architecture cognitive. Elle permet, entre autres (Kotseruba et Tsotsos, 2020) :

— De stocker les résultats intermédiaires de calculs ;
— L'apprentissage ;
— L'adaptation à l'environnement changeant.

Toutefois, les implémentations particulières des systèmes de mémoires diffèrent d'un système à un autre.

Dans les architectures cognitives, la mémoire à court terme est généralement décomposée en (Kotseruba et Tsotsos, 2020) :

— Mémoire sensorielle, qui stocke et prétraite les perceptions récentes ;
— Mémoire de travail, qui contient les éléments liés à la tâche en cours d'exécution.

La mémoire à long terme, quant à elle, est généralement décomposée en (Kotseruba et Tsotsos, 2020) :

— Mémoire procédurale, qui stocke les informations sur les actions à entreprendre dans certaines conditions. Elle est non-déclarative et contient les connaissances implicites ;
— Mémoire sémantique, qui stocke les connaissances factuelles ;
— Mémoire épisodique, qui stocke les épisodes de l'expérience personnelle du système, lesquels seront activés en cas de présence d'une situation similaire. Elle est dite autobiographique.

Les mémoires sémantique et épisodique sont déclaratives et contiennent les connaissances explicites.

Dans les architectures cognitives symboliques, la connaissance procédurale est représentée par un ensemble de règles « si … alors » préprogrammées ou apprises pour un domaine particulier.

Certaines architectures cognitives présentent une mémoire unifiée, c'est-à-dire qu'aucune distinction n'est faite entre la mémoire à court terme et la mémoire à long terme. Ce type de mémoire est appelé mémoire globale (Kotseruba et Tsotsos, 2020).

1.2.2.3.5 L'apprentissage

L'apprentissage désigne la capacité d'un système à améliorer ses performances au fil du temps. Il repose sur l'expérience et correspond à l'acquisition de connaissances. On distingue deux grands types d'apprentissage (Squire, 1992) :

- L'apprentissage déclaratif ou explicite. Ici, la connaissance est déclarative et est un ensemble de faits sur le monde et de diverses relations définies entre eux.
- L'apprentissage non déclaratif. Celui-ci comprend plusieurs sous-types : L'apprentissage perceptif, qui découle de la perception ; l'apprentissage procédural, qui concerne l'acquisition de compétences pratiques ; l'apprentissage associatif, qui implique des processus de décision influencés par des récompenses et des sanctions (comme l'apprentissage par renforcement ou l'apprentissage hebbien) ; l'apprentissage non associatif, où il n'existe pas de lien direct entre les stimuli et les réponses (comme dans l'apprentissage par accoutumance ou par habituation).

1.2.2.3.6 Le raisonnement

Le raisonnement est la capacité à traiter les connaissances de manière logique et systématique. Il influence et structure toutes les formes d'activité humaine (Osta-Vélez, 2014).

Plusieurs types de raisonnement sont distingués, notamment : la déduction, l'induction, l'abduction, l'heuristique, l'analogique, le narratif, le moral, etc. (Kotseruba et Tsotsos, 2020).

Dans les architectures cognitives, le but du raisonnement est de déterminer la meilleure action à entreprendre et de l'exécuter.

1.2.2.3.7 La métacognition

La métacognition, telle que définie par Flavell (1979), désigne l'aptitude à « penser sur sa propre pensée ». Elle regroupe un ensemble de compétences permettant d'observer, d'analyser et de réguler ses propres processus cognitifs de manière introspective. Elle est utilisée dans l'apprentissage et la prise de décision.

Les trois mécanismes métacognitifs les plus courants sont :

- l'auto-observation,
- l'auto-analyse,
- l'auto-régulation.

1.2.2.4 Les applications pratiques des architectures cognitives

Les architectures cognitives sont utilisées dans l'ensemble des domaines où des logiciels doivent, à l'aide d'une entrée de données, effectuer une action cohérente visant à optimiser un processus.

Dix grandes catégories d'applications pratiques ont été identifiées (Kotseruba et Tsotsos, 2020) :

- Les expériences psychologiques (IRM, EEG, etc.) : L'architecture cognitive ACT-R a été utilisée pour simuler les régions cérébrales activées lors de tâches de mémoire, permettant de comparer les résultats avec des données d'IRM fonctionnelle ou d'EEG (Anderson *et al.*, 1997).
- La robotique : Des architectures cognitives comme LIDA ou DIARC ont été utilisées pour piloter des robots capables de percevoir leur environnement, prendre des décisions et interagir avec des humains, comme dans le cas des robots d'assistance domestique (Scheutz et Schermerhorn, 2011).
- Les jeux et les puzzles : Les architectures cognitives SOAR et CLARION ont été utilisées pour résoudre des puzzles comme les Tours de Hanoï ou pour apprendre à jouer à des jeux comme le Tetris ou le Sudoku, en reproduisant les stratégies humaines (Sun *et al.*, 2001).
- La modélisation des performances humaines : Des architectures cognitives comme EPIC ont été employées pour simuler le comportement d'opérateurs humains dans des environnements critiques (simulateurs d'avion) afin d'évaluer la charge cognitive et la réactivité (Kieras et Meyer, 1997).
- Le traitement du langage naturel : L'architecture cognitive NL-Soar a été développée pour comprendre et générer du langage naturel, intégrant des capacités de traitement en temps réel et de dialogue (Rubinoff et Lehman, 1994).
- Les interactions homme–robot : L'architecture cognitive DIARC a été utilisée pour doter les robots

de capacités de dialogue, de reconnaissance des émotions et d'adaptation à des comportements humains dans un cadre collaboratif (Brick et Scheutz, 2007).

— Les interactions homme–ordinateur : L'architecture cognitive EPIC a été utilisée pour modéliser les performances humaines dans des environnements multitâches, notamment pour évaluer et améliorer la conception des interfaces utilisateur dans des contextes tels que les cockpits d'avions (Kieras et Meyer, 1997).
— La vision assistée par ordinateur : Des architectures cognitives intégrant des modules de perception visuelle (comme ACT-R couplé à une vision par ordinateur) ont permis à un système de détecter et d'interpréter des éléments dans une scène visuelle (Gray et Fu, 2004).
— La catégorisation et le regroupement (reconnaissance des formes, etc.) : L'architecture cognitive CLARION a été utilisée pour modéliser l'apprentissage de catégories (animaux, outils) à partir d'exemples perceptifs, mimant les mécanismes humains d'abstraction (Sun, 2006).
— Les agents virtuels et diverses applications : Des architectures cognitives comme MIDCA ont permis de créer des agents intelligents dans des environnements simulés (militaires, aériens, etc.) pour entraîner ou tester des comportements sans risque réel (Cox *et al.*, 2016).

1.3 Quelques théories cognitives majeures et leurs applications dans la modélisation de la cognition humaine

Au cours des quatre dernières décennies, la recherche sur les architectures cognitives n'a cessé de croître. En 2018, le nombre d'architectures existantes avait atteint plusieurs centaines, dont quarante-neuf étaient encore activement développées pour un ensemble de disciplines allant de la psychanalyse aux neurosciences (Kotseruba et Tsotsos, 2020). Pour une revue approfondie des architectures cognitives, voir (Kotseruba et Tsotsos, 2020), qui présente un panorama de 85 architectures cognitives.

Plusieurs théories cognitives majeures ont été proposées dans la littérature pour expliquer le fonctionnement de l'esprit humain, parmi lesquelles la théorie des processus duaux, qui distingue les modes de pensée automatique et contrôlé, et la théorie de Baars, qui met l'accent sur le rôle de la conscience dans l'intégration des informations à travers différents systèmes cognitifs. À celles-ci s'ajoute la théorie unifiée de la cognition, proposée par Allen Newell, qui cherche à rendre compte de l'ensemble des processus cognitifs au sein d'un cadre théorique unique, en considérant l'esprit comme un système cohérent et intégré.

— La théorie des processus duaux englobe l'ensemble des processus cognitifs, allant de la perception

sensorielle aux opérations mentales de haut niveau, telles que la résolution de problèmes (Kahneman, 2011). Selon cette théorie, il existe chez les humains deux systèmes de raisonnement distincts : le système S1 (ou Type 1), qui est inconscient, spontané, rapide, parallèle et approximatif, et le système S2 (ou Type 2), qui est conscient, acquis, lent, sériel et rigoureux (Kahneman, 2011; Evans et Stanovich, 2013). Le système S1 a un fonctionnement biologique, tandis que le système S2 a un fonctionnement plutôt formel.

— La théorie de l'espace de travail global, ou théorie de Baars (Baars, 1988), propose un cadre pour comprendre la conscience et les processus cognitifs conscients. Développée principalement par Bernard Baars en 1988, cette théorie suggère que la conscience résulte de l'intégration et de la diffusion d'informations à travers un « espace de travail global » dans le cerveau. Elle a inspiré Stan Franklin dans le développement d'un agent intelligent pour la marine américaine, appelé IDA (Intelligent Distribution Agent) (Franklin, 2003).

— La théorie unifiée de la cognition a été principalement développée par Allen Newell. Dans son ouvrage *Unified Theories of Cognition* (Newell, 1990), il propose une approche intégrée visant à expliquer le fonctionnement de la cognition humaine. Newell soutient que l'esprit fonctionne comme un système unifié, et que les modèles cognitifs doivent être capables de rendre compte de l'ensemble des comportements cognitifs, tels que la résolution de problèmes, la prise de décision, la perception ou encore l'apprentissage. Il met en avant l'architecture cognitive Soar comme une implémentation concrète de cette théorie, capable de générer des sous-objectifs et d'apprendre de manière continue à partir de l'expérience. Dans cette même perspective, les travaux de John R. Anderson s'inspirent de la théorie cognitive de Newell et postulent que la cognition résulte de l'interaction entre plusieurs modules spécialisés, chacun étant responsable d'un aspect spécifique du traitement de l'information, comme la mémoire déclarative, la mémoire procédurale, la perception ou encore l'attention. Bien qu'initialement fondée sur la théorie unifiée de Newell, l'architecture ACT-R évolue progressivement, au fil des recherches, vers une théorie de plus en plus complète de la cognition humaine (Anderson, 2007; Raluca, 2013).

Plusieurs agents cognitifs ont été implémentés dans des thèses de doctorat en informatique cognitive de l'UQAM, dans le but de simuler l'esprit humain et de reproduire des capacités d'apprentissage similaires à celles de l'humain. Parmi ces travaux, on peut citer :

— Sébastien Hélie (2007) propose une théorie unificatrice de la cognition humaine, expliquant à la fois

l'apprentissage ascendant (des connaissances implicites aux explicites) et l'interaction synergique entre ces deux types de connaissances. Il développe une architecture cognitive appelée TELECAST (TEnsor LEarning of CAusal STructure) pour implémenter cette théorie, capable de reproduire des phénomènes cognitifs observés chez les humains.

— Daniel Dubois (2007) propose une architecture d'agent tutoriel qui s'inspire de théories cognitives comme celle de Baars et d'architectures cognitives telles qu'ACT-R. Le système développé, nommé CTS (Conscious Tutoring System), est basé sur IDA mais adapté aux besoins du tutorat.
— Usef Faghihi (2011) explore le rôle des émotions dans l'apprentissage et leur intégration dans un agent cognitif. Il présente un agent cognitif conscient appelé CELTS (Conscious Emotional Learning Tutoring System), doté d'émotions et de plusieurs mécanismes d'apprentissage, tels que l'apprentissage émotif, épisodique, procédural et causal. CELTS a été utilisé comme module pédagogique intelligent dans CanadarmTutor, un système tutoriel développé pour la formation des astronautes à la manipulation du bras robotique Canadarm2. Cet agent, basé sur des théories cognitives, simule les interactions qu'un instructeur humain aurait avec un apprenant (Faghihi *et al.*, 2013).
— Othalia Larue (2015) propose une architecture cognitive basée sur la théorie des processus duaux, plus précisément le modèle tripartite de Stanovich, capable de rendre compte de la dualité du comportement humain, où les comportements réactifs (rapides, automatiques et intuitifs) et délibératifs (lents, réfléchis et contrôlés) coexistent et contribuent de manière fluide au fonctionnement d'un système unifié.

Il convient de rappeler que l'objectif de notre recherche n'est pas de développer une nouvelle architecture cognitive, mais d'utiliser une architecture existante et éprouvée pour l'appliquer au domaine de l'aviation, afin de reproduire les processus cognitifs d'un pilote humain. Nous avons sélectionné quelques architectures cognitives parmi celles qui nous ont semblé pertinentes pour réaliser notre projet. Celles que nous décrivons dans ce document sont : ACT-R (1973), SOAR (1983), CLARION (1994), SPAUN (2013) et LEABRA (2016).

1.4 ACT-R (Adaptive Control of Thought – Rational) (Anderson *et al.*, 2004; Anderson, 2007)

1.4.1 Description

La famille d'architectures ACT, développée par Anderson et ses collègues, est la plus ancienne (depuis 1973) et de loin la plus connue, avec le plus grand nombre d'articles scientifiques consacrés à son étude (Kotseruba

et Tsotsos, 2020). L'objectif initial d'ACT-R est la description d'une théorie complète de la cognition humaine.

ACT-R est un système de production composé d'une mémoire déclarative ainsi que d'une mémoire procédurale.

- La première mémoire contient des connaissances telles que « Yaoundé est la capitale du Cameroun », « le Cameroun est un pays d'Afrique » ou encore « $8 + 2 = 10$ ».
- La seconde contient les règles de production, qui représentent les connaissances sur la façon d'effectuer certaines tâches comme jouer aux échecs, conduire une voiture ou réaliser une opération arithmétique.

Le module déclaratif est responsable de la reconnaissance des informations présentées au modèle et du calcul de l'activation des règles, tandis que la partie procédurale détermine l'utilité des règles activées et déclenche la plus appropriée. La cognition émerge ainsi de l'interaction entre ces structures procédurales et déclaratives.

La cognition repose sur des modules, chacun dédié au traitement d'une information particulière. ACT-R comprend notamment :

- un module responsable de l'identification des objets présents dans le champ visuel ;
- un module moteur/manuel ;
- un module déclaratif pour la récupération des informations ;
- un module responsable du maintien de l'objectif et des intentions actuelles.

Les modules sont coordonnés par un module central capable de reconnaître des patrons d'informations présents dans les buffers (tampons) et de les modifier. Ce module recherche dans la mémoire procédurale une règle de production correspondant à l'état actuel des tampons. Une seule règle de production peut être exécutée à un moment donné. Une fois exécutée, elle peut modifier les tampons et ainsi changer l'état du système.

Ainsi, la cognition se déroule comme une succession de déclenchements de règles de production. La sélection de la règle de production à exécuter repose sur l'apprentissage par l'utilité, qui choisit la règle ayant la plus grande utilité. L'apprentissage dans le module procédural de l'ACT-R utilise une équation de différence simple, similaire à la règle d'apprentissage de Rescorla-Wagner et à la règle delta de Widrow et Hoff.

L'information est encapsulée dans ces modules, qui ne communiquent que via les buffers. Le fonctionnement des modules se fait en parallèle et de manière asynchrone.

On distingue deux unités de connaissances : les chunks (unités de la mémoire déclarative, contenant des connaissances factuelles) et les règles de production (unités de la mémoire procédurale).

En plus des structures symboliques précédemment décrites, ACT-R possède un niveau sous-symbolique qui affecte la sélection des chunks (en influençant les niveaux d'activation) et des règles de production (en y associant des utilités espérées utilisables en cas de conflit) ainsi que l'apprentissage. Deux méthodes d'apprentissage sous-symbolique sont utilisées : l'apprentissage par renforcement et l'apprentissage statistique.

Par exemple, ACT-R peut être utilisé pour simuler une tâche de conditionnement classique dans laquelle un rat apprend à appuyer sur un levier pour obtenir de la nourriture. Si un levier est détecté par le module déclaratif (stimulus), toutes les règles associées dans la mémoire procédurale sont activées. Si l'une d'elles indique qu'appuyer sur le levier fait apparaître de la nourriture, et que l'organisme modélisé a faim, l'utilité de cette règle sera élevée (calculée par le module procédural). ACT-R choisira alors d'appuyer sur le levier.

ACT-R est une architecture cognitive hybride (Kotseruba et Tsotsos, 2020), dont la structure et les mécanismes d'apprentissage en font un modèle puissant pour la simulation de comportements humains dans divers domaines. Son architecture, tirée de (Raluca, 2013), est présentée à la Figure 1.2 ci-dessous.

1.4.2 Applications et domaines d'utilisation

ACT-R est flexible et offre un temps de réaction réduit. Il a été utilisé avec succès pour créer des modèles dans des domaines tels que :

- L'apprentissage et la mémoire ;
- La résolution de problèmes et la prise de décision ;
- Le langage et la communication ;
- La perception et l'attention ;
- Le développement cognitif, etc.

Outre ses applications en psychologie cognitive, ACT-R a été utilisé dans les domaines suivants :

- L'interaction homme-machine, pour produire des modèles d'utilisateurs permettant d'évaluer diffé-

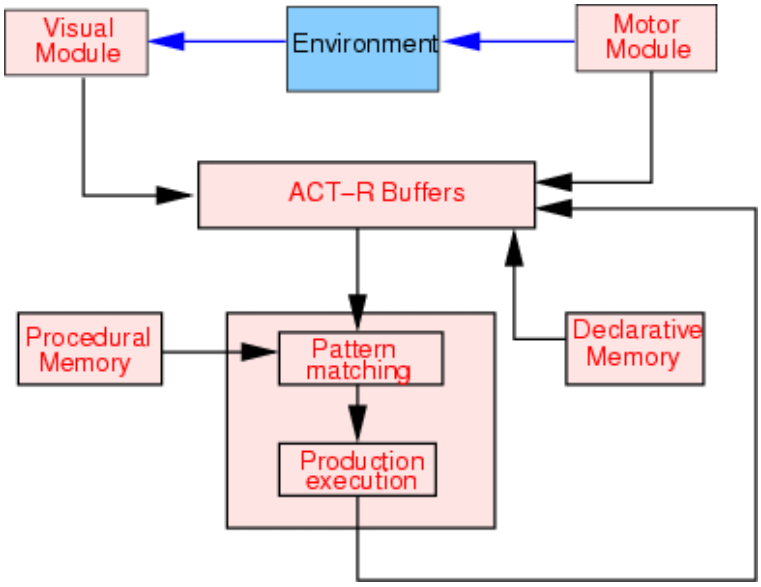


Figure 1.2 – Architecture de ACT-R

rentes interfaces informatiques ;

— L'éducation (systèmes de tutorat cognitif), pour deviner les difficultés que peuvent rencontrer les étudiants et leur fournir une aide ciblée ;

— La modélisation des performances humaines, pour fournir des agents cognitifs qui habitent les environnements de formation ;

— La neuropsychologie, pour interpréter les données de l'IRMf (Imagerie par Résonance Magnétique Fonctionnelle) ;

— L'aviation, etc.

ACT-R a aussi été utilisé dans les projets suivants :

— Jeu logique (tic-tac-toe, tour de Hanoï) ;

— Commande de dîner par téléphone ;

— Déplacement dans un édifice ;

— Prise de décision au croisement d'une route ;

— Aide à la conduite d'une voiture ;

— Reconnaissance des animaux selon certaines caractéristiques ;

— Apprentissage d'une langue étrangère ;

— Modélisation d'un comportement humain dans un jeu ;

— Les tuteurs cognitifs, etc.

1.5 SOAR (State, Operator And Result) (Laird, 2012; Newell, 1990)

1.5.1 Description

SOAR, qui signifie *State, Operator And Result* (Laird, 2012; Newell, 1990), est la théorie unifiée de la cognition proposée par Newell. Toutes les tâches, de la plus simple (routine) à la plus complexe, y sont représentées sous forme d'espaces problèmes, et la prise de décision se fait selon l'hypothèse de l'Espace Problème.

Les espaces problèmes sont définis comme un ensemble d'états et d'opérateurs qui manipulent ces états. Les états finaux sont ceux qui mèneront à la résolution du problème (ou de la tâche). SOAR contient une liste de buts emmagasinés dans une pile (goal stock), gérée à l'aide d'une mémoire de travail et d'une mémoire à long terme. La mémoire de travail sert à :

- identifier les stimuli présents ;
- récupérer les règles de production ayant ces stimuli parmi leurs prémisses (les règles étant emmagasinées dans la mémoire à long terme).

Ensuite, l'utilité de chaque règle présente dans la mémoire de travail est évaluée en fonction de la pile de buts, et la règle la plus appropriée est choisie.

Tout comme ACT-R, SOAR construit des *chunks* pour automatiser l'utilisation des règles les plus courantes, permettant ainsi l'acquisition de nouvelles connaissances permanentes, constituant la mémoire épisodique du système.

SOAR est aussi doté de capacités motrices et sensorielles. Le système perceptuel génère des éléments qui peuvent être stockés temporairement en mémoire à court terme.

SOAR supporte actuellement plusieurs mécanismes d'apprentissage, tels que :

- le *chunking* ;
- l'apprentissage par renforcement ;
- l'apprentissage sémantique et épisodique.

Il dispose également de différents systèmes de mémoires à long terme, dont :

- procédural ;

— épisodique ;
— sémantique.

SOAR utilise aussi diverses représentations, telles que :

— symbolique ;
— numérique ;
— visuelle.

L'imagerie mentale visuelle est présente dans SOAR, grâce à des processus qui créent des structures symboliques à partir des images. Cela permet de résoudre des problèmes de raisonnement spatial. Une mémoire à court terme est associée à l'imagerie mentale visuelle, laquelle a accès à une mémoire à long terme des images et peut ainsi construire et manipuler des images.

Enfin, SOAR a été étendu pour inclure des émotions qui interviennent au niveau de l'apprentissage.

De nos jours, SOAR est une architecture cognitive hybride (Kotseruba et Tsotsos, 2020), et sa représentation schématique, tirée de (Chong *et al.*, 2009) est donnée à la Figure 1.3 ci-dessous.

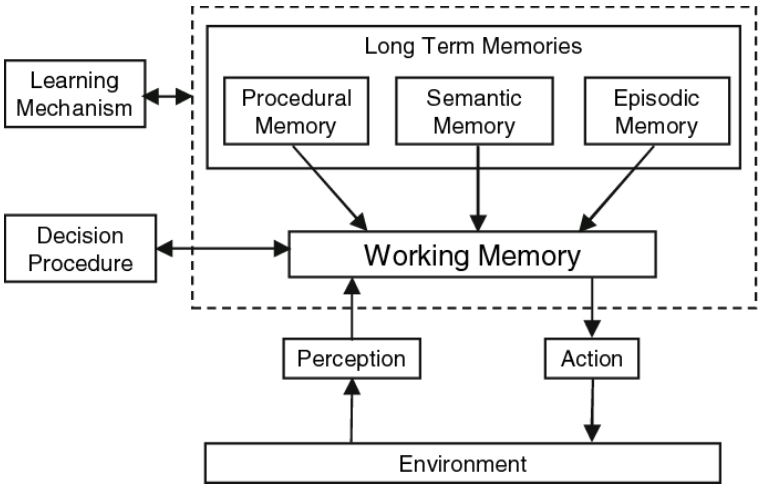


Figure 1.3 – Architecture de SOAR

1.5.2 Applications et domaines d'utilisation

SOAR a connu un succès notable dans une variété de domaines, apportant des avancées significatives telles que :

— L'intégration de la mémoire de travail et de la mémoire épisodique, ainsi que leurs interactions avec les autres types de mémoires.
— Une meilleure coordination entre les différents types de mémoires (procédurale, sémantique, épisodique) et leurs fonctions respectives.
— La prise en compte des émotions dans les processus d'apprentissage, permettant une modélisation plus réaliste des comportements cognitifs.

Par ailleurs, SOAR a été appliqué avec succès dans des domaines variés, notamment :

— Les jeux stratégiques, où il excelle dans la planification et l'exécution de tactiques complexes.
— La planification et la résolution de problèmes, en optimisant les processus décisionnels dans des environnements contraints.
— La recherche en neurosciences, où il sert de cadre théorique pour étudier les mécanismes de la mémoire et de l'apprentissage.

Récemment, dans le domaine de la sécurité informatique, SOAR a démontré son utilité dans plusieurs applications clés (Dilmegani et Palazoğlu, 2025), telles que :

— La détection et la réponse aux attaques de phishing.
— La gestion des incidents de sécurité, en automatisant les réponses aux menaces.
— La gestion des vulnérabilités, en identifiant et en priorisant les failles à corriger.
— L'orchestration de la sécurité dans le cloud, en centralisant et en coordonnant les outils de protection.

1.6 CLARION (Connectionist Learning with Adaptive Rule Induction ON-line) (Sun, 2002, 2016)

1.6.1 Description

L'architecture cognitive CLARION (Connectionist Learning with Adaptive Rule Induction ON-line) est également très connue. C'est une architecture qui combine des processus psychologiques implicites et explicites. L'ambition des recherches autour de CLARION est principalement de comprendre les mécanismes de la cognition humaine (prise de décision, apprentissage, raisonnement, etc.), tout en développant également des agents artificiels.

Une caractéristique importante de CLARION est la distinction entre les processus implicites et explicites,

ainsi que l'accent mis sur la capture de l'interaction entre ces deux types de processus.

CLARION est composée de quatre sous-systèmes distincts, avec une double structure de représentation dans chaque sous-système (représentations implicites et explicites), à savoir :

- Le sous-système centré sur l'action, dont le rôle est de contrôler les actions externes et internes.
- Le sous-système non centré sur l'action, dont le rôle est de maintenir les connaissances générales.
- Le sous-système motivationnel, qui fournit des motivations sous-jacentes pour la perception, l'action et la cognition.
- Le sous-système méta-cognitif, dont le rôle est de surveiller, diriger et modifier les opérations des autres sous-systèmes.

Les actions du sous-système méta-cognitif comprennent :

- La fixation d'objectifs pour le sous-système centré sur l'action ;
- La définition de paramètres pour les sous-systèmes centré sur l'action et non-action ;
- La modification d'un processus en cours dans les sous-systèmes centrés sur l'action et la non-action.

CLARION est une architecture cognitive hybride (Kotseruba et Tsotsos, 2020), combinant des représentations connexionnistes et des représentations symboliques, dont l'architecture, tirée de (Chong *et al.*, 2009), est donnée à la Figure 1.4 ci-dessous.

1.6.2 Applications et domaines d'utilisation

Puisque CLARION est un modèle reposant en partie sur l'apprentissage par renforcement, il est parfaitement adapté à l'apprentissage de séquences d'événements. Cette architecture a été utilisée dans plusieurs domaines, démontrant sa polyvalence et sa capacité à modéliser des processus cognitifs complexes. On peut citer :

- La navigation et la résolution de problèmes spatiaux : CLARION a été appliqué avec succès pour naviguer dans des environnements complexes, tels que des labyrinthes et des champs de mines.
- La modélisation de la mémoire implicite : CLARION a été utilisé pour simuler des tâches comme l'apprentissage de séquences motrices (par exemple, taper sur un clavier) ou la reconnaissance de motifs, où la mémoire implicite joue un rôle clé.
- Apprentissage et adaptation dynamique : Dans des environnements changeants, CLARION a démon-

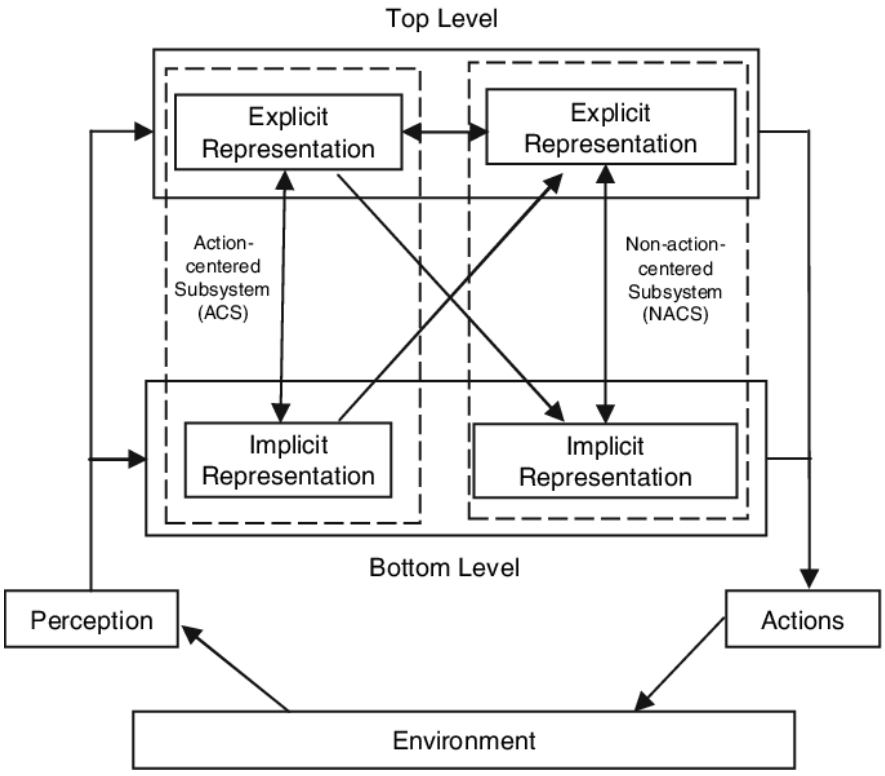


Figure 1.4 – Architecture de CLARION

tré sa capacité à ajuster ses stratégies en fonction des nouvelles informations, ce qui est utile dans des domaines comme la robotique autonome ou les systèmes de recommandation.

— Applications en IA : Grâce à son architecture hybride, CLARION a été utilisé pour développer des agents intelligents capables de raisonner, d'apprendre et de s'adapter.

1.7 SPAUN (Semantic Pointer Architecture Unified Network) (Eliasmith, 2012; Stewart *et al.*, 2012; Eliasmith, 2013)

1.7.1 Description

SPAUN (*Semantic Pointer Architecture Unified Network*) est une architecture cognitive qui se définit comme un modèle informatique du cerveau humain, construit par le professeur Chris Eliasmith et ses collègues de l'Université de Waterloo, au Canada (Eliasmith, 2012).

Il comprend environ deux millions et demi de neurones virtuels organisés en sous-systèmes qui ressemblent à des régions spécifiques du cerveau, telles que :

— le cortex préfrontal ;

— les ganglions de la base ;
— le thalamus.

À titre de comparaison, le cerveau humain contient environ cent milliards de neurones (Grillner et Robertson, 2015).

SPAUN est capable de (Stewart *et al.*, 2012) :

— reconnaître des nombres et s'en souvenir ;
— comprendre des séquences numériques et les écrire à l'aide d'un bras robotique ;
— répondre à des questions sur les nombres et compléter des séries numériques après avoir observé des exemples.

Bien qu'il possède des capacités avancées de perception, de reconnaissance et d'exécution des comportements requis, SPAUN reste lent comparé au cerveau humain. En effet, chaque seconde biologique nécessite environ deux heures de traitement informatique.

Une caractéristique remarquable de SPAUN est qu'il présente des limitations similaires à celles des humains. Par exemple :

— il a du mal à mémoriser des listes de chiffres trop longues ;
— il se souvient mieux des premiers et derniers éléments d'une liste (effet de primauté et de récence) ;
— il manifeste une hésitation avant de répondre aux questions, comme les humains, ce qui pourrait améliorer l'interaction avec les futurs robots en les rendant plus naturels et intuitifs à utiliser.

Enfin, SPAUN est le premier simulateur de cerveau capable d'accomplir une série de tâches variées et de démontrer des comportements complexes.

Bien que basé sur l'architecture SPA (Semantic Pointer Architecture) qui s'inscrit dans le paradigme connexionniste (Kotseruba et Tsotsos, 2020), SPAUN constitue une architecture cognitive hybride qui intègre à la fois des représentations naturalistes et des représentations symboliques (Eliasmith, 2012), en utilisant des pointeurs sémantiques (vecteurs) pour encoder des informations symboliques de haut niveau et des réseaux de neurones biologiquement plausibles pour le traitement parallèle et distribué.

L'architecture de SPAUN, tirée de (Blouw *et al.*, 2016b), est illustrée à la Figure 1.5 ci-dessous.

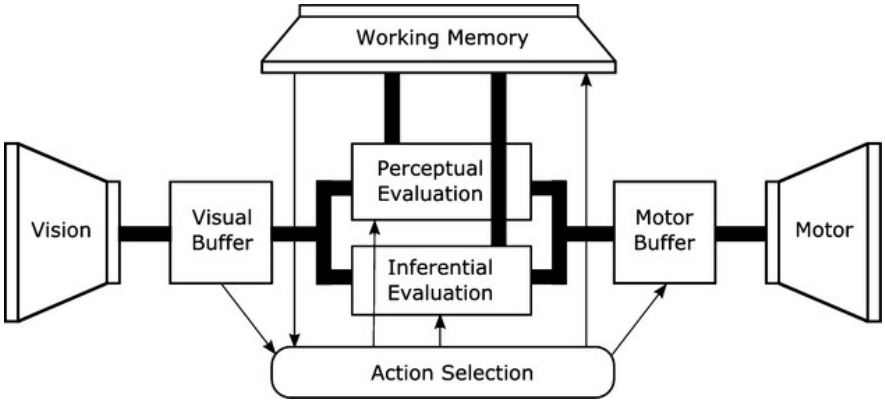


Figure 1.5 – Architecture de SPAUN.

1.7.2 Applications et domaines d'utilisation

SPAUN ressemble davantage au cerveau humain que les modèles précédents, et il pourrait donc être utilisé dans l'étude de certains troubles cérébraux (Eliasmith, 2012). Dans une expérience en 2012, par exemple, il a examiné les effets de la mort des neurones dans une simulation du vieillissement du cerveau humain, permettant aux chercheurs d'observer la dégradation progressive des performances cognitives, similaire à celle observée dans des maladies neurodégénératives comme Alzheimer ou Parkinson (Eliasmith, 2012).

Le modèle pourrait également être utile pour le développement d'applications d'IA multitâches et de robotique, notamment dans la création de systèmes capables d'adaptation contextuelle et d'apprentissage continu (Blouw *et al.*, 2016a). SPAUN se distingue par sa capacité à intégrer la reconnaissance visuelle, la mémoire de travail et la prise de décision dans une architecture unifiée, ce qui en fait un candidat prometteur pour des applications nécessitant une intelligence cognitive intégrée, telles que les assistants personnels avancés, les systèmes de diagnostic médical ou les interfaces homme-machine intuitives (Stewart et Eliasmith, 2014).

De plus, SPAUN offre un cadre précieux pour la recherche fondamentale en neurosciences computationnelles, permettant de tester des hypothèses sur le fonctionnement du cerveau humain sans les contraintes éthiques liées aux expérimentations sur des sujets humains (Kajić *et al.*, 2017). Cette caractéristique en fait un outil potentiel pour le développement de thérapies neurologiques innovantes et la modélisation des effets de différents médicaments sur les fonctions cognitives (Choo et Eliasmith, 2013).

Dans le domaine éducatif, SPAUN pourrait contribuer à la création de systèmes d'apprentissage adaptatifs qui s'ajustent au profil cognitif de chaque apprenant, en imitant la façon dont le cerveau humain intègre et traite de nouvelles informations (Rasmussen *et al.*, 2017).

1.8 LEABRA (Learning in an Error-driven and Associative, Biologically Realistic Algorithm) (O'Reilly, 1996; O'Reilly *et al.*, 2017)

1.8.1 Description

LEABRA, acronyme de *Learning in an Error-driven and Associative, Biologically Realistic Algorithm*, est une architecture cognitive développée en 2016. Il s'agit d'un modèle d'apprentissage qui combine les principes de l'apprentissage associatif de type Hebbien et ceux de l'apprentissage basé sur l'erreur. Cette approche repose sur un réseau neuronal artificiel conçu pour reproduire de manière réaliste certaines propriétés des réseaux neuronaux biologiques. Ce modèle permet de prédire les réponses produites par le réseau en fonction des données d'entrée et de l'expérience acquise lors des apprentissages précédents.

LEABRA s'inscrit dans une perspective cohérente avec les avancées actuelles en neurosciences. Son architecture repose sur trois niveaux hiérarchiques, chacun correspondant à une échelle distincte d'organisation du traitement cognitif :

- Niveau micro : À ce niveau, les connexions entre neurones sont caractérisées par des poids synaptiques, et chaque neurone est doté d'une fonction d'activation régulant sa réponse aux stimuli reçus.
- Niveau méso : Ce niveau met en œuvre un équilibre entre l'apprentissage associatif de type Hebbien et l'apprentissage par essais et erreurs. L'apprentissage repose également sur des mécanismes d'inhibition ou de renforcement des connexions synaptiques.
- Niveau macro : L'architecture modélise les interactions fonctionnelles entre plusieurs régions cérébrales, notamment le cortex postérieur (traitement sensoriel), l'hippocampe (mémoire épisodique), les ganglions de la base (émotion, via l'amygdale, etc.) et le cortex frontal (prise de décision et contrôle moteur).

LEABRA est souvent perçu comme un système hybride, principalement connexionniste (O'Reilly *et al.*, 2017). L'architecture combine des principes biologiquement inspirés tout en offrant des capacités cognitives avancées, généralement attribuées aux systèmes symboliques. Elle est neurocomputationnelle (Kotseruba et Tsotsos, 2020), et son architecture de haut niveau, tirée de (O'Reilly *et al.*, 2017), est représentée schéma-

tiquement à la Figure 1.6 ci-dessous.

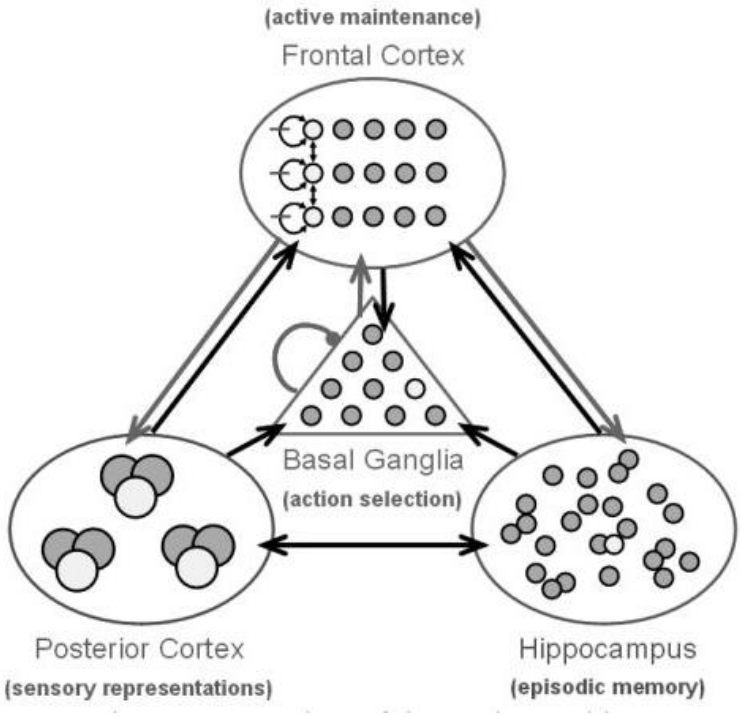


Figure 1.6 – Architecture de LEABRA.

1.8.2 Applications et domaines d'utilisation

LEABRA offre une meilleure implémentation des rôles respectifs des différentes mémoires à long terme et de leur fonctionnement (mémoires procédurale, épisodique et sémantique) ainsi qu'une meilleure implémentation des différentes procédures d'apprentissage. D'après O'Reilly *et al*. (2017), le modèle LEABRA a trouvé des applications dans plusieurs domaines :

- Apprentissage et mémoire : LEABRA modélise les mécanismes d'apprentissage supervisé et non supervisé, en intégrant des principes d'apprentissage par erreur et associatif pour reproduire la manière dont le cerveau humain acquiert et stocke des informations.
- Perception et reconnaissance : L'architecture a été utilisée pour simuler des processus perceptifs, notamment la reconnaissance d'objets et de formes, en reproduisant les dynamiques des aires visuelles du cortex.
- Fonctions exécutives et prise de décision : En modélisant les interactions entre le cortex préfrontal et les ganglions de la base, LEABRA aide à comprendre comment le cerveau gère la planification, le

contrôle cognitif et la prise de décisions.

— Mémoire épisodique : LEABRA simule les fonctions de l'hippocampe pour expliquer comment les souvenirs d'événements spécifiques sont encodés, consolidés et rappelés.

1.9 Démarche de sélection de l'architecture cognitive

1.9.1 Propriétés recherchées

L'architecture cognitive que nous recherchons doit répondre à plusieurs critères essentiels pour le développement d'un pilote synthétique fondé sur une architecture cognitive utilisant les ontologies comme base de connaissances. Les propriétés sont présentées ci-dessous

— Démontrer l'exécution d'une tâche complexe dans un contexte particulier : L'architecture doit être capable de démontrer son efficacité en exécutant des tâches complexes dans un environnement réaliste.

— Avoir été utilisée avec succès dans l'aviation : Pour garantir sa fiabilité, son applicabilité, sa pertinence et sa validité dans le domaine de l'aviation, l'architecture cognitive doit avoir fait ses preuves dans ce secteur, démontrant ainsi sa capacité à gérer des tâches critiques et spécifiques au pilotage.

— Gérer les mémoires déclaratives et procédurales de façon distincte : L'ontologie de référence repose sur ces deux types de connaissances. L'architecture doit donc être capable de gérer ces deux types de mémoires de manière distincte pour refléter fidèlement le fonctionnement cognitif humain.

— Avoir un module de sélection automatique de la règle de production à exécuter : Pour éviter les conflits lors du choix des procédures à exécuter, l'architecture doit inclure un module capable de sélectionner automatiquement et de manière transparente la règle de production appropriée.

— Avoir une architecture de fonctionnement hybride : L'architecture doit être capable d'utiliser simultanément des processus symboliques et sous-symboliques, essentiels pour modéliser avec précision les comportements cognitifs.

— Prendre en compte les émotions dans son raisonnement : Pour refléter les dimensions affectives du pilote humain, l'architecture cognitive doit intégrer la gestion des émotions dans son processus de raisonnement, permettant ainsi de modéliser l'impact des émotions sur l'exécution des tâches de pilotage.

— Permettre l'intégration d'ontologies comme base de connaissances : Afin d'assurer une représentation structurée et formelle des connaissances, l'architecture cognitive doit être capable d'intégrer des ontologies. Celles-ci serviront à modéliser les connaissances nécessaires à l'exécution des tâches

complexes de pilotage, ainsi qu'à formaliser l'environnement d'exécution de ces tâches.

— Avoir une implémentation en Python : Pour assurer une communication et une intégration efficace avec les autres modules du système global, qui sont principalement écrits en Python, il est crucial que l'architecture cognitive soit compatible avec ce langage.

1.9.2 Analyse comparative des architectures candidates

À partir des huit propriétés de l'architecture cognitive que nous recherchons, présentées à la section 1.9.1, nous établissons dans le tableau 1.2 le classement des architectures cognitives candidates sur lesquelles se fondera notre pilote synthétique.

Critères / Architectures	**ACT-R**	**SOAR**	**CLARION**	**SPAUN**	**LEABRA**
Exécution de tâches complexes	Oui	Oui	Oui	Oui	Oui
Utilisation dans l'aviation	Oui	Oui	Non	Non	Non
Gestion distincte des mémoires déclaratives et procédurales	Oui	Oui	Oui	Non	Non
Module de sélection automatique des règles de production	Oui	Oui	Oui	Non	Non
Architecture hybride (symbolique et sous-symbolique)	Oui	Oui	Oui	Oui	Oui
Prise en compte des émotions	Oui	Oui	Non	Oui	Oui
Intégration d'ontologies	Oui	Oui	Non	Non	Non
Implémentation en Python	Oui	Oui	Oui	Oui	Non
Total (Nombre de « oui »)	**8**	**8**	5	4	3

Table 1.2 – Analyse comparative des architectures cognitives candidates selon les propriétés recherchées.

D'après le tableau 1.2, ACT-R et SOAR, ayant obtenu un score de huit sur huit, sont les seules architectures cognitives répondant à l'ensemble des propriétés recherchées.

1.9.3 Justification du choix d'ACT-R

Pour départager ACT-R et SOAR, nous avons comparé, à travers le tableau 1.3, chacune des huit propriétés de ces deux architectures et avons retenu celle qui a obtenu le meilleur score dans cette analyse.

Critères / Architectures	**ACT-R**	**SOAR**	**Gagnant**
Exécution de tâches complexes	Capable d'exécuter des tâches complexes dans divers domaines	Capable d'exécuter des tâches complexes dans divers domaines	Égalité
Utilisation dans l'aviation	Utilisé dans des simulations de pilotage et des études sur la performance des pilotes	Utilisé dans des systèmes de simulation de vol et d'entraînement des pilotes	Égalité
Gestion distincte des mémoires déclaratives et procédurales	Distinction claire entre mémoire déclarative (*chunks*) et mémoire procédurale (règles de production)	Distinction entre mémoire sémantique, épisodique, procédurale, mais on y rencontre le plus souvent des goulots d'étranglement dans ces mémoires.	ACT-R
Module de sélection automatique des règles de production	Utilise un système d'activation et de correspondance par pattern avec des mécanismes de conflit basés sur l'utilité	Utilise un mécanisme de sélection d'opérateurs avec préférences et l'apprentissage par *chunking*	Égalité
Architecture hybride (symbolique et sous-symbolique)	Fortement hybride avec des représentations symboliques et des processus sous-symboliques (activation, utilité) pour modéliser le raisonnement humain	Principalement symbolique, avec des éléments sous-symboliques limités introduits dans les versions récentes	ACT-R

Critères / Architectures	ACT-R	SOAR	Gagnant
Prise en compte des émotions	Intègre partiellement les émotions via des modules spécifiques	Intégration limitée des émotions dans les versions standard	Égalité
Intégration d'ontologies	Permet l'intégration d'ontologies comme base de connaissances	Supporte l'utilisation d'ontologies pour structurer les connaissances	Égalité
Implémentation en Python	Plusieurs implémentations Python disponibles comme `pyACT-R`, `CCMSuite`	Moins d'implémentations Python robustes disponibles	ACT-R
Gagnant	**3**	**0**	**ACT-R**

Table 1.3 – Analyse comparative des architectures cognitives ACT-R et SOAR selon les propriétés recherchées.

D'après le tableau 1.3, ACT-R arrive en tête du classement, ce qui motive notre choix en faveur de cette architecture cognitive.

En effet, dans ACT-R, les mémoires procédurales et déclaratives sont distinctes. De plus, cette architecture intègre et gère efficacement les principales fonctions cognitives, notamment la mémoire, l'apprentissage, la perception et l'action, tout en tenant compte de leurs limitations (Jones *et al.*, 2007; Anderson *et al.*, 2004). Par ailleurs, la règle de production la plus pertinente est sélectionnée de manière transparente et dynamique (Raluca, 2013).

ACT-R bénéficie d'une large compatibilité avec plusieurs langages de programmation, notamment Lisp, Java, C++ et Python, à travers divers packages qui permettent d'émuler ses mécanismes. De plus, cette architecture, aujourd'hui considérée comme hybride, a été largement utilisée avec succès dans plusieurs domaines, notamment l'aviation. Elle a notamment servi à modéliser des activités de conduite, qu'il s'agisse de la conduite automobile (Salvucci, 2006) ou du pilotage d'aéronefs, en simulant le comportement cognitif d'un agent dans un contexte de pilotage (Gluck *et al.*, 2003). Enfin, ACT-R dispose de plusieurs implémentations en Python, dont pyACT-R (Brasoveanu et Dotlačil, 2020), que nous utiliserons dans notre travail.

1.10 Conclusion

L'exploration des architectures cognitives ACT-R, SOAR, CLARION, SPAUN et LEABRA révèle la richesse et la complexité des efforts visant à modéliser l'intelligence humaine. Chaque architecture représente une perspective unique sur la cognition, offrant des enseignements précieux dans la compréhension des processus mentaux. Les points clés de notre analyse peuvent être résumés comme suit :

- Diversité des approches : Ces architectures illustrent la variété des approches pour modéliser la cognition, allant des modèles symboliques aux modèles connexionnistes, en passant par les approches hybrides.
- Capacités cognitives fondamentales : Toutes ces architectures visent à reproduire des capacités essentielles telles que la perception, la mémoire, l'apprentissage, le raisonnement et la prise de décision.
- Réalisme et complexité : Chaque modèle tente de capturer différents aspects du réalisme cognitif, qu'il soit psychologique, biologique ou écologique.
- Limitations : Malgré leurs avancées significatives, ces architectures présentent encore des limitations importantes, notamment en termes de vitesse de traitement, de généralisation et de reproduction exacte des processus cognitifs humains.
- Potentiel interdisciplinaire : Ces architectures démontrent l'immense potentiel des approches interdisciplinaires, combinant des connaissances en psychologie cognitive, neurosciences, IA et philosophie de l'esprit.

Ces architectures offrent un cadre pour la simulation de comportements complexes, allant de la perception, de la mémoire à la prise de décision et à l'apprentissage. Leur applicabilité dans des domaines variés, notamment l'aviation, démontre leur potentiel pour la conception d'un pilote synthétique capable de reproduire les processus cognitifs des pilotes humains.

Pour notre projet de pilote synthétique, ces architectures offrent un riche réservoir de méthodes et de concepts. Elles suggèrent qu'une approche hybride, intégrant des éléments symboliques et connexionnistes, pourrait être la plus prometteuse pour capturer la complexité du comportement cognitif d'un pilote humain.

Par ailleurs, l'analyse comparative réalisée nous a conduit à sélectionner ACT-R, qui répond à l'ensemble des critères requis pour la modélisation d'un pilote synthétique. Cette architecture offre notamment une

gestion distincte des mémoires déclarative et procédurale, des mécanismes d'apprentissage adaptés et une capacité d'intégration des ontologies, caractéristiques qui en font une solution robuste pour simuler les comportements et décisions d'un pilote humain.

La prochaine étape consistera à adapter ACT-R au contexte du pilotage aérien en y intégrant une base de connaissances ontologique spécifiquement conçue pour représenter les procédures, les concepts et les contraintes de l'aviation. Cette approche nous permettra d'élaborer un modèle cognitif capable de reproduire fidèlement les mécanismes de prise de décision et les réactions d'un pilote humain en situation de vol.

CHAPITRE 2
FONDEMENTS THÉORIQUES DU PILOTE SYNTHÉTIQUE : CHOIX DE L'ONTOLOGIE ET DE L'ARCHITECTURE COGNITIVE

2.1 Introduction

L'aviation moderne repose sur des systèmes technologiques avancés visant à améliorer constamment la sécurité et l'efficacité des opérations aériennes. Dans ce contexte évolutif, l'émergence de technologies cognitives comme le pilote synthétique représente une avancée significative dans la compréhension et la modélisation des processus mentaux des pilotes.

Ce chapitre explore les fondements théoriques qui sous-tendent le développement d'un pilote synthétique basé sur l'architecture cognitive ACT-R qui intègre une ontologie de référence. Nous examinerons les composantes essentielles qui permettront de créer un agent capable de reproduire les mécanismes cognitifs complexes d'un pilote humain.

Notre exploration débutera par une analyse détaillée du pilote automatique, système technologique fondamental dans l'aviation moderne, qui a révolutionné les opérations de vol en automatisant des tâches répétitives. Nous mettrons en lumière son fonctionnement, ses avantages et son rôle crucial dans l'assistance aux équipages.

Nous nous pencherons ensuite sur le concept de pilote synthétique, et plus particulièrement sur l'implémentation basée sur une architecture cognitive. Cette approche vise à modéliser non seulement les actions d'un pilote, mais aussi ses processus de raisonnement, ses mécanismes de prise de décision et sa capacité d'adaptation en situations complexes.

Un accent particulier sera mis sur la modélisation cognitive et l'assistance cognitive dans un cockpit, démontrant comment les architectures cognitives peuvent transformer notre compréhension des interactions homme-machine dans des environnements critiques comme l'aviation.

Nous proposerons également un état de l'art des ontologies ayant été intégrées dans des agents cognitifs appliqués au domaine de l'aviation, afin de mettre en lumière les travaux existants et de situer notre

approche dans la continuité de ces recherches.

Aussi, nous détaillerons le modèle de référence ontologique, outil conceptuel permettant de formaliser les connaissances nécessaires à l'exécution de tâches complexes de pilotage. Nous expliciterons ses composantes fondamentales, son architecture et son potentiel pour la formalisation des connaissances dans le domaine du pilotage.

Nous procéderons ensuite à une comparaison détaillée des architectures cognitives présentées au chapitre1 pour sélectionner celle qui correspond le mieux aux besoins de notre projet. Cette analyse nous conduira à choisir l'architecture cognitive ACT-R, reconnue pour sa capacité à modéliser de manière réaliste et efficace les processus cognitifs des pilotes humains.

De même, nous retiendrons le modèle de référence ontologique comme l'ontologie sur laquelle reposera notre pilote synthétique, en raison de sa capacité à formaliser les connaissances spécifiques à l'aviation.

Enfin, nous montrerons comment l'intégration d'un modèle ontologique qui formalise les procédures de pilotage d'un avion dans une architecture cognitive avancée comme ACT-R, peut ouvrir de nouvelles perspectives pour le développement de systèmes d'assistance cognitive dans le domaine aéronautique.

2.2 Pilote automatique : Définition, fonctionnement et avantages

2.2.1 Définition

De façon générale, un pilote automatique est un dispositif de conduite automatique d'un véhicule, permettant de contrôler la trajectoire sans intervention humaine constante. Il est utilisé dans divers véhicules tels que les avions, les bateaux et les véhicules spatiaux (Federal Aviation Administration, 2014).

Dans l'aviation, le pilote automatique est un système de contrôle automatisé qui guide un aéronef sans intervention constante du pilote humain (Harris, 2011; Duperrin, 2021). Formellement défini comme un système qui exécute automatiquement les fonctions attribuées par le pilote, dans les limites opérationnelles prédéfinies du système (Federal Aviation Administration, 2014), il constitue un élément fondamental de l'avionique moderne.

Selon Billings (1996), le pilote automatique représente l'une des premières formes d'automatisation aéro-

nautique qui a révolutionné le secteur de l'aviation en permettant des vols plus longs et plus stables tout en réduisant la charge de travail des équipages. Son invention date de 1912 par Lawrence Burst Sperry (Scheck, 2017).

2.2.2 Fonctionnement

Le fonctionnement du pilote automatique repose sur plusieurs sous-systèmes intégrés qui travaillent en synergie. Les systèmes modernes peuvent gérer des phases complexes du vol, y compris l'atterrissage automatique (Duperrin, 2021). D'après Wickens *et al.* (1998), le pilote automatique utilise des capteurs pour collecter des données sur l'altitude, la vitesse, la direction et d'autres paramètres de vol. Ces informations sont ensuite traitées par un ordinateur central qui compare les valeurs actuelles aux valeurs de consigne définies par les pilotes.

Le système est capable de contrôler (Duperrin, 2021; Aprendamos Aviación, 2022) :

- la gouverne de profondeur, permettant de contrôler l'altitude de l'avion ;
- la gouverne de direction, qui permet de contrôler la trajectoire de l'avion ;
- les ailerons qui ont pour rôle le contrôle du roulis de l'avion ;
- la vitesse de l'avion.

Le pilote automatique fonctionne selon un principe de boucle fermée où les écarts entre l'état souhaité et l'état réel sont constamment mesurés et corrigés (Sarter et Woods, 1995).

Un système de pilote automatique est parfois appelé familièrement *George*. Les pilotes disent souvent : *Nous laisserons George voler pendant un moment*. L'origine du surnom reste incertaine : certains affirment qu'il fait référence à l'inventeur George De Beeson, qui a breveté un pilote automatique dans les années 1930, tandis que d'autres estiment que les pilotes de la Royal Air Force l'ont adopté pendant la Seconde Guerre mondiale en référence au roi George VI (Baker, 2020; Van, 2014).

2.2.3 Avantages

Le pilote automatique offre de nombreux avantages qui expliquent son utilisation généralisée dans l'aviation moderne (Wiener et Curry, 1980; Sarter et Woods, 1995; Wickens *et al.*, 1998; Dekker, 2014; Wickens *et al.*, 2021) :

— Réduction de la charge de travail : En automatisant les tâches de pilotage basiques, le pilote automatique permet aux pilotes de se concentrer sur d'autres aspects critiques comme la navigation, la communication et la gestion des systèmes.
— Précision accrue : Le système informatique peut maintenir des paramètres de vol avec une précision supérieure à celle d'un pilote humain, particulièrement durant les vols de longue durée.
— Économie de carburant : Une trajectoire plus stable et optimisée permet de réduire la consommation de carburant.
— Sécurité renforcée : Le pilote automatique aide à prévenir la fatigue des pilotes et peut maintenir un vol stable dans des conditions difficiles (turbulences, mauvaise visibilité).
— Capacités étendues : Le pilote automatique permet des opérations spécifiques comme les atterrissages par faible visibilité.

2.3 Pilote synthétique : Définition, fonctionnement et avantages

2.3.1 Définition

Un pilote synthétique est un agent logiciel conçu pour simuler les comportements, les décisions et les actions d'un pilote humain dans un environnement aéronautique (Cacciabue, 1998). Contrairement au pilote automatique classique, le pilote synthétique ne se limite pas à reproduire les actions d'un pilote humain, mais cherche également à imiter ses processus de raisonnement. Son objectif est d'assister et de compléter les pilotes humains, plutôt que de les remplacer entièrement (Boy, 2017).

D'après Guy André Boy (2017), un pilote synthétique est capable d'apprendre des interactions avec les pilotes humains et de s'adapter à leurs préférences et styles de pilotage. En d'autres termes, le pilote synthétique constitue une tentative de recréer, par des moyens informatiques, les processus décisionnels et comportementaux d'un pilote réel face à diverses situations de vol (Corker et Smith, 1993).

D'après les travaux de Foyle et Hooey (2007), les pilotes synthétiques peuvent être classés en deux catégories principales suivant leur approche de modélisation du comportement humain :

— Les modèles basés sur des règles simples et des algorithmes déterministes : Ils suivent un ensemble de règles prédéfinies pour prendre des décisions ou effectuer des actions. Ces modèles fonctionnent selon des principes fixes, où chaque entrée produit une sortie spécifique de manière prévisible, sans variation ni adaptation aux conditions changeantes.

— Les modèles fondés sur des architectures cognitives complètes : Ils simulent les processus mentaux humains.

Le pilote synthétique fondé sur une architecture cognitive comme ACT-R ou SOAR, représente une implémentation spécifique du concept de pilote synthétique utilisant cette architecture cognitive. Autrement dit, c'est un modèle computationnel qui simule les processus cognitifs d'un pilote humain en se basant sur les principes théoriques et les mécanismes de l'architecture cognitive en question (Ritter *et al.*, 2018). Il joue un rôle de soutien au pilotage, modélisant ainsi l'esprit d'un pilote humain en pensant et agissant comme lui.

2.3.2 Fonctionnement

Dans cette section, nous décrirons le fonctionnement d'un pilote synthétique basé sur l'architecture cognitive ACT-R. Le fonctionnement d'un pilote synthétique utilisant une autre architecture cognitive serait similaire, car il reposera sur les composants fondamentaux de cette architecture.

Le fonctionnement du pilote synthétique ACT-R s'articule autour de plusieurs composants clés (Ritter *et al.*, 2018; Anderson *et al.*, 2004; Anderson, 2007) :

— Le module perceptuel-moteur : Il gère les interactions avec l'environnement, y compris la vision, l'audition et les actions motrices. Dans le contexte du pilotage, il simule la façon dont un pilote perçoit les instruments et manipule les commandes.
— Le module déclaratif : Il représente la mémoire à long terme du pilote, contenant des connaissances factuelles sur les procédures de vol, les caractéristiques de l'aéronef et les règles de navigation aérienne. Ce module utilise un mécanisme d'activation qui simule les effets de récence et de fréquence sur la récupération des souvenirs.
— Le module procédural : Il contient des règles de production de la forme « si … alors » qui déterminent les actions à entreprendre dans différentes situations. Ce module utilise, entre autres, un mécanisme d'apprentissage par renforcement qui ajuste l'utilité des règles en fonction des résultats obtenus.
— Le module but : Il maintient une représentation hiérarchique des objectifs du pilote, depuis les buts généraux (comme « effectuer un atterrissage ») jusqu'aux sous-objectifs spécifiques (comme « sortir le train d'atterrissage »).

— Le module imaginal : Il simule la mémoire de travail spatiale, particulièrement importante pour maintenir une représentation mentale de la situation de vol et de la position de l'aéronef.
— Les mécanismes d'apprentissage : Ils reposent sur le renforcement des règles efficaces, l'affaiblissement des associations peu utilisées et la compilation de nouvelles règles à partir de séquences existantes.

2.3.3 Avantages

Les pilotes synthétiques présentent plusieurs avantages parmi lesquels (Salas *et al.*, 2010; Oliver *et al.*, 2019) :

— Une modélisation réaliste du comportement humain, permettant des simulations plus précises.
— Une amélioration de la formation des pilotes, en leur permettant d'interagir avec un agent cognitif proche d'un pilote humain.
— Une assistance en temps réel, en suggérant des actions adaptées aux situations rencontrées.
— Une gestion des erreurs humaines, en identifiant et en corrigeant des déviations par rapport aux procédures standard.
— Un comportement adaptatif, en adaptant des stratégies en fonction de l'évolution des conditions de vol, au lieu de suivre des procédures fixes.
— Un entraînement et une évaluation améliorés, en permettant de simuler des interactions pour l'entraînement des équipages et l'évaluation des nouvelles technologies de cockpit.
— Un apprentissage incrémental, en intégrant des mécanismes d'apprentissage qui permettent au pilote synthétique d'améliorer ses performances avec l'expérience, de manière analogue à un pilote humain.

Alors que le pilote automatique est déjà intégré aux avions et à certains véhicules, le pilote synthétique demeure un domaine de recherche en développement. Ses bénéfices potentiels pour l'aviation en font un dispositif prometteur, justifiant ainsi l'orientation de nos travaux de recherche.

2.4 Modélisation et assistance cognitive dans un cockpit

2.4.1 Modélisation cognitive

La modélisation cognitive est une approche qui se situe à l'intersection de plusieurs disciplines, dont la psychologie, les neurosciences et l'informatique, dans le but de comprendre et de simuler les processus mentaux humains à travers des modèles informatiques. Elle s'intéresse à l'étude et à la reproduction de fonctions cognitives essentielles, telles que la perception, l'attention, la mémoire et la prise de décision (Strube, 2001; Byrne *et al.*, 2004; FasterCapital, 2024). Elle permet aussi de démontrer comment les gens s'y prennent pour résoudre des problèmes et accomplir des tâches (IxDF, 2020).

La représentation du comportement individuel des utilisateurs est un défi pour la modélisation cognitive. La plupart des modèles vise à simuler le comportement moyen des utilisateurs dans des conditions contrôlées plutôt que la performance individuelle dans des tâches complexes (Rehling *et al.*, 2004). La représentation du comportement individuel dans des contextes naturalistes exige de traiter de multiples sources de variation telles que les différences interindividuelles (Taatgen, 1999) et les facteurs externes non contrôlés de la situation. Par exemple, lors de la modélisation de la performance des pilotes dans l'aviation commerciale, il faut tenir compte des différents niveaux d'expérience des pilotes, des conditions météorologiques changeantes et de l'état de l'aéronef.

2.4.2 Assistance cognitive dans un cockpit

Un agent intelligent, s'appuyant sur un modèle cognitif capable de suivre à la fois le contexte opérationnel et la dynamique cognitive d'un utilisateur, peut constituer une base solide pour l'assistance cognitive dans les opérations (Zhang *et al.*, 2018). Dans ce cas, l'assistance cognitive consiste à fournir la bonne information au bon moment. La qualité de l'assistance qui peut être fournie dépend donc de ce qui est et peut être connu de l'environnement de la tâche, des processus cognitifs, affectifs et comportementaux de l'opérateur.

L'assistance cognitive désigne un ensemble de technologies conçues pour soutenir les capacités mentales d'une personne en fournissant des informations, des suggestions ou des rappels au moment approprié. L'objectif est d'améliorer la performance dans des tâches complexes en réduisant les impacts de certaines limites humaines, comme la surcharge cognitive ou les erreurs d'inattention, et en facilitant la prise de décision dans des environnements changeants et exigeants. Elle peut se manifester sous diverses formes, telles que des systèmes de soutien à la prise de décision, des rappels visuels ou auditifs, ou encore des

agents intelligents qui interagissent avec l'utilisateur pour l'aider à accomplir une tâche de manière plus efficace. Dans des domaines tels que l'aviation, la médecine ou le domaine militaire, l'assistance cognitive est particulièrement importante pour garantir la sécurité et la performance dans des environnements à haut risque et à forte charge cognitive (Estes *et al.*, 2016; Zhang *et al.*, 2018; Gagnon-Roy *et al.*, 2024).

Dans le cockpit, la surdité d'inattention chez les pilotes entraîne une baisse de performance (Dehais *et al.*, 2019). Ils peuvent alors bénéficier d'une assistance cognitive, par exemple sous forme de rappels verbaux ou visuels (Estes *et al.*, 2016). Les causes d'inattention peuvent être diverses et dépendre de certains facteurs tels que les facteurs perceptifs, attentionnels, émotionnels, etc. (Dehais *et al.*, 2019). Les alertes sont déclarées comme manquées lorsque les pilotes réagissent mal ou bien, ne réagissent pas du tout. Savoir ce qui a fait qu'un pilote n'a pas bien réagi ou, quels éléments d'information il n'a pas pu traiter donne une valeur de diagnostic et aide à identifier les moyens de soutien adéquats.

Pour l'assistance cognitive dans le traitement des alertes du poste de pilotage, les informations sur le contenu d'un message et sur son traitement par le pilote constituent une alternative viable aux modèles complexes nécessaires à la prédiction déterministe des états des utilisateurs. Les conséquences d'un message entendu ou ignoré sur les performances des pilotes peuvent être anticipées à l'aide d'un modèle cognitif de pilote.

Nous présentons également l'assistance cognitive à travers plusieurs exemples utilisant ACT-R, une architecture cognitive complète et scientifiquement validée, qui a permis de modéliser une large gamme de processus cognitifs. ACT-R a notamment été employé pour simuler le contrôle manuel d'un avion monomoteur (Somers et West, 2013), l'allocation de l'attention visuelle dans un cockpit (Byrne *et al.*, 2004), ainsi que l'acquisition et l'utilisation de compétences liées au système de gestion de vol (Schoppek et Boehm-Davis, 2004; Taatgen *et al.*, 2008). Les modèles ACT-R développés dans ces études offrent des représentations formelles des tâches et des processus impliqués dans le vol, facilitant ainsi la définition de critères de performance normative. Dans (Oliver *et al.*, 2020), un modèle ACT-R neuroadaptatif s'est montré plus précis pour représenter les réponses des pilotes individuels aux alertes et aux messages que le système conventionnel (87% de précision contre 72%).

2.5 Quelques travaux sur l'intégration d'ontologies dans des agents cognitifs appliqués à l'aviation

Sans viser l'exhaustivité, nous proposons ici une sélection de travaux fondateurs ayant contribué à l'évolution des architectures cognitives intégrant des ontologies dans des agents cognitifs, en lien avec des applications spécifiques au domaine de l'aviation. Ces contributions ont joué un rôle clé dans la structuration de notre démarche de recherche.

Wu *et al.* (2014) proposent une approche basée sur une ontologie pour modéliser les aéronefs et améliorer les systèmes de gestion de maintenance. Leur recherche se concentre sur l'utilisation des ontologies comme fondement d'un système cognitif de gestion de maintenance aéronautique plus efficace. En développant une modélisation ontologique des composants d'aéronefs, de leurs relations et de leurs caractéristiques, les auteurs permettent une meilleure classification des pannes, une planification optimisée de la maintenance et une réduction des coûts opérationnels. Cette approche facilite également l'intégration des données historiques de maintenance avec les connaissances expertes des techniciens aéronautiques.

Insaurralde et Blasch (2016) suggèrent une représentation des connaissances fondée sur une ontologie pour soutenir la prise de décision dans les systèmes avioniques. Leur approche vise à intégrer des données hétérogènes provenant de capteurs et de systèmes embarqués, en les structurant dans une ontologie qui permet une interprétation contextuelle et une prise de décision plus rapide et plus précise. Cette méthode est particulièrement utile dans des environnements complexes où la rapidité et la fiabilité des décisions sont critiques, comme dans les situations d'urgence ou de gestion du trafic aérien.

Yousefzadeh Aghdam *et al.* (2021) développent une ontologie des messages de sécurité aérienne afin d'améliorer l'analyse des risques et la formation des contrôleurs. Leur ontologie permet une classification et une interprétation structurée des messages de sécurité, facilitant ainsi l'identification des risques potentiels et la mise en place de mesures préventives. Cette approche contribue à améliorer la sécurité aérienne en fournissant aux contrôleurs un outil puissant pour comprendre et réagir aux situations critiques.

Dimosthenis Stefanidis *et al.* (2020) présentent l'ontologie ICARUS, une ontologie générale de l'aviation adoptant une approche multicouche. L'ontologie ICARUS est exploitée via des requêtes SPARQL, permettant une interrogation flexible et efficace des données aéronautiques. Cette ontologie couvre un large éventail de concepts liés à l'aviation, offrant une base de connaissances robuste pour des applications variées, allant de la gestion du trafic aérien à la maintenance des aéronefs.

Insaurralde et Blasch (2018, 2019, 2022) proposent une approche fondée sur une ontologie pour la prise de décision avancée en gestion du trafic aérien, intégrant une architecture cognitive. Leur travaux se concentrent sur la modélisation des incertitudes et des dynamiques complexes du trafic aérien, en utilisant des ontologies pour structurer les connaissances et faciliter la prise de décision en temps réel. Cette approche est particulièrement utile pour améliorer la perception de la situation des contrôleurs et des pilotes, en leur fournissant des outils cognitifs pour interpréter et réagir aux situations complexes.

Dario Salvucci (2006) utilise l'architecture cognitive ACT-R pour modéliser le comportement des conducteurs, une approche potentiellement applicable aux pilotes d'aéronefs. Bien que centrée sur la conduite automobile, cette recherche démontre la capacité des architectures cognitives à modéliser des comportements humains complexes dans des tâches de contrôle, ouvrant la voie à des applications similaires dans le domaine de l'aviation.

Oliver *et al*. (2019) exploitent ACT-R pour modéliser les processus décisionnels des pilotes en situation d'urgence. Leur travail se concentre sur la simulation des processus cognitifs impliqués dans la gestion des alertes dans le cockpit, contribuant ainsi à une meilleure compréhension des facteurs humains dans les situations critiques de vol.

Oltramari et Lebiere (2011) proposent une approche intégrant les ontologies dans l'architecture cognitive ACT-R pour la résolution de problèmes. Cette approche combine la représentation formelle des connaissances offerte par les ontologies avec les capacités de simulation cognitive d'ACT-R, ouvrant des perspectives pour la modélisation du comportement des pilotes et la conception de systèmes d'aide à la décision plus performants.

Les recherches précédentes ont posé des bases importantes pour l'intégration des ontologies dans des agents cognitifs dans l'aviation. Elles ont abordé divers aspects tels que la maintenance des aéronefs, la gestion du trafic aérien, la sécurité et la formation. Cependant, ces études n'ont pas exploité le potentiel d'une expertise des procédures de pilotage formalisée dans une ontologie. Le modèle de référence ontologique répond à cette préoccupation.

2.6 Le modèle de référence ontologique (Courtemanche *et al.*, 2022, 2023; Ghaderi *et al.*, 2022b)

Le modèle de référence ontologique est une structure conceptuelle qui repose sur deux composantes principales : l'ontologie du domaine et l'ontologie des tâches. Ces deux éléments sont étroitement liés et permettent de formaliser les connaissances nécessaires à l'exécution des tâches complexes dans un environnement spécifique, en l'occurrence le pilotage d'un aéronef. Le modèle vise à décomposer les tâches en règles exécutables tout en intégrant les contraintes et les paramètres de l'environnement d'exécution.

2.6.1 L'ontologie du domaine

L'ontologie du domaine est un ensemble terminologique spécifique à l'environnement d'exécution des tâches de pilotage. Elle sert de dictionnaire des données, décrivant les éléments de l'environnement intérieur (poste de pilotage), extérieur (conditions météorologiques, espace aérien) et les systèmes de l'aéronef. Cette ontologie est structurée en classes générales et spécialisées, permettant une représentation cohérente et modulaire des paramètres nécessaires à l'exécution des tâches. L'ontologie du domaine est organisée autour d'une classe générale « Environment » qui se divise en trois grandes catégories : « AircraftInsideEnvironment », « AircraftOutsideComponent » et « AircraftSystems ». La description de ces classes est la suivante :

- Environment (Environnement) : Il s'agit de la classe générale autour de laquelle est construite l'ontologie du domaine. Elle est la généralisation des environnements intérieur, extérieur et des systèmes de l'aéronef.
- AircraftInsideEnvironment (Environnement intérieur) : Elle regroupe l'ensemble des éléments retrouvés à l'intérieur du poste de pilotage.
- AircraftOutsideComponent (Environnement extérieur) : Elle représente les éléments de l'environnement extérieur.
- AircraftSystems (Systèmes) : Elle englobe les systèmes de l'aéronef qui n'appartiennent pas directement à l'environnement intérieur ou extérieur.

La Figure 2.1, adaptée de (Courtemanche *et al.*, 2023), fournit une vue d'ensemble de l'ontologie du domaine sous forme graphique.

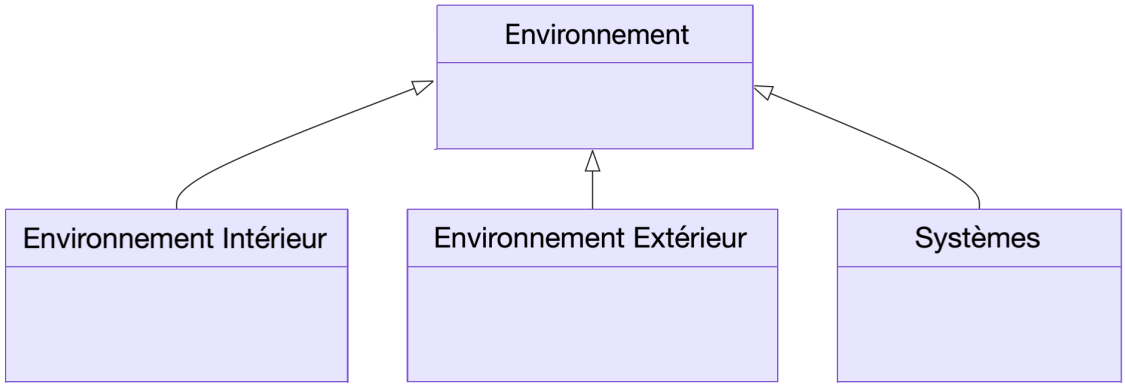


Figure 2.1 – L'ontologie du domaine.

2.6.2 L'ontologie des tâches

L'ontologie des tâches constitue l'élément central du modèle de référence. Elle structure le processus de résolution de problèmes en décomposant les tâches complexes en sous-tâches tout en intégrant les pré-conditions et contraintes propres au contexte de pilotage.

Chaque tâche est ainsi divisée en sous-tâches ordonnées selon des préconditions spécifiques. Par exemple, une tâche de décollage peut inclure la vérification des instruments, l'accélération sur la piste et la rotation de l'appareil. Ces sous-tâches sont associées à des paramètres précis de l'ontologie du domaine. Par exemple, la vérification des instruments peut être reliée à la classe « Environnement intérieur » pour accéder aux données des instruments de bord.

L'ontologie des tâches et l'ontologie du domaine sont étroitement interconnectées par des liens sémantiques. Chaque contrainte définie dans l'ontologie des tâches identifie un paramètre de l'environnement ainsi qu'un intervalle de valeurs acceptables. De même, l'exécution des tâches repose sur ces liens, en spécifiant les paramètres pertinents dans l'ontologie du domaine et en déterminant les valeurs à leur attribuer.

Enfin, cette ontologie est conçue pour être interprétée par un moteur de règles, permettant une exécution automatique des tâches en fonction des conditions de l'environnement.

La Figure 2.2, adaptée de (Courtemanche *et al.*, 2023), illustre visuellement l'ontologie des tâches. Chacun des termes utilisés pour décomposer les procédures est défini de manière concise :

— Task (Tâche) : Il s'agit de l'élément central de l'ontologie des tâches, où l'information et les para-

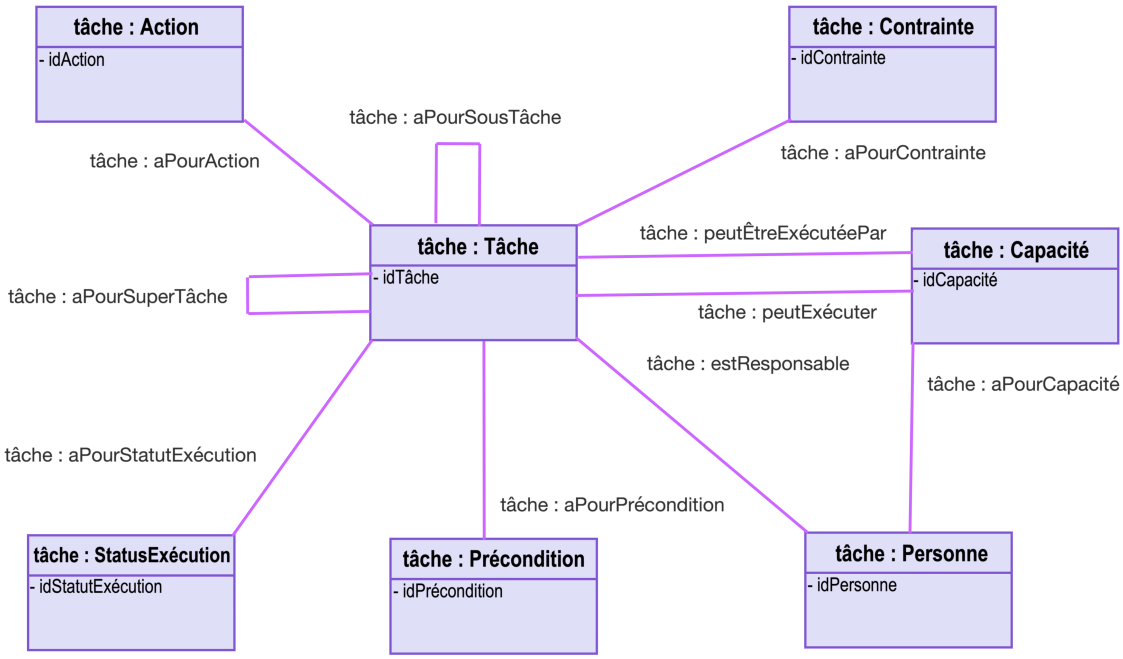


Figure 2.2 – L'ontologie des tâches.

mètres d'exécution sont organisés en fonction de leurs relations avec les autres classes du modèle. Chaque tâche a un identifiant unique (ID) et est liée à des paramètres spécifiques.

— Capability (Capacité) : Dans une situation de pilotage, les rôles de pilote en action (pilot flying) et de pilote en surveillance (pilot monitoring) sont attribués indépendamment des titres de pilote et copilote. Chaque tâche doit être exécutée par l'un de ces deux rôles, et cette classe précise quel type d'individu doit accomplir chaque tâche spécifique.

— Person (Personne) : Cette classe distingue le pilote du copilote et précise le rôle de chacun en fonction du contexte. Elle associe une responsabilité d'exécution, que ce soit pour le pilote en action (pilot flying) ou en surveillance (pilot monitoring), qui peut être soit le commandant de bord, soit le copilote.

— Precondition (Précondition) : Cet élément définit les tâches qui doivent être accomplies avant de pouvoir exécuter d'autres tâches. Il permet de capturer l'ordre chronologique dans lequel les tâches doivent être réalisées.

— ExecutionStatus (État d'exécution) : Cet élément indique si une tâche spécifique a été accomplie ou non. Il sert à suivre l'avancement du processus d'exécution.

— Action (Action) : En lien étroit avec l'ontologie du domaine, cet élément spécifie l'action à appliquer

sur l'environnement. Il définit à la fois le paramètre concerné et la valeur à imposer.

— Constraint (Contrainte) : L'exécution d'une tâche est soumise à des conditions spécifiques de l'environnement. Cette notion précise le paramètre et la valeur qui doivent être atteints pour permettre l'exécution. Les contraintes peuvent prendre différentes formes, comme une valeur exacte, une valeur minimale ou une valeur maximale. Combinées aux préconditions, elles créent un lien sémantique fort avec l'environnement, limitant l'exécution à des paramètres précis.
— Attention (Attention) : Chaque tâche est associée à un score d'attention, reflétant la charge cognitive nécessaire à son exécution. Ce niveau d'attention est ancré sémantiquement dans le modèle d'exécution. En reliant ces données au modèle, on offre un moyen efficace de surveiller la charge cognitive pendant l'exécution automatique des procédures.

2.7 Analyse critique et proposition de solutions

2.7.1 Propriétés de l'ontologie recherchée

L'ontologie recherchée doit répondre à plusieurs exigences clés pour soutenir la modélisation cognitive et opérationnelle du pilote synthétique, notamment :

— Formalisation des connaissances : L'ontologie doit permettre de formaliser les connaissances nécessaires à l'exécution des tâches complexes dans un environnement de pilotage d'un aéronef, en structurant les informations de manière claire et exploitable.
— Formalisation de l'environnement d'exécution des tâches : L'ontologie doit également formaliser l'environnement dans lequel les tâches de pilotage sont exécutées, permettant ainsi une modélisation précise des interactions entre le pilote synthétique et son environnement opérationnel.
— Interopérabilité avec l'architecture cognitive : L'ontologie doit être conçue pour s'intégrer de manière fluide avec l'architecture cognitive choisie, en fournissant une base de connaissances cohérente et accessible pour les processus cognitifs modélisés.
— Extensibilité et modularité : L'ontologie doit être conçue de manière à être extensible et modulaire, permettant ainsi d'ajouter de nouvelles connaissances ou de modifier les structures existantes sans compromettre l'intégrité globale du système.
— Capacité à gérer des concepts complexes : L'ontologie doit être capable de représenter des concepts complexes et leurs relations, en particulier ceux liés aux tâches de pilotage, afin de soutenir un raisonnement approfondi et contextuel.

2.7.2 Justification du choix de l'ontologie de référence pour notre étude

La revue de la littérature sur l'intégration des ontologies dans des architectures cognitives en aviation, présentée à la section 2.5, a mis en évidence la contribution significative de ces travaux à la formalisation des connaissances dans divers domaines de l'aéronautique. Ces contributions couvrent des aspects allant de la maintenance des aéronefs à la gestion du trafic aérien, en passant par la sécurité et la formation des agents. Cependant, bien que ces recherches aient jeté des bases solides pour l'utilisation des ontologies dans des architectures cognitives en aviation, aucune n'a pleinement exploité le potentiel de l'expertise aéronautique formalisée dans une ontologie pour offrir une assistance cognitive aux pilotes, que ce soit pendant leur formation ou lors de la prise de décision opérationnelle.

Par ailleurs, le modèle de référence ontologique formalise non seulement l'expertise des pilotes, mais aussi les connaissances liées à l'environnement dynamique de l'aéronef. Il possède ainsi les propriétés essentielles d'une ontologie de référence adaptées à notre approche, comme détaillé à la section 2.7.1. Nous l'appellerons dans la suite du document « ontologie de référence ».

Enfin, notre pilote synthétique reposera sur l'architecture cognitive ACT-R et s'appuiera sur l'ontologie de référence décrite à la section 2.6 comme base de connaissances. Celle-ci sera ensuite enrichie par l'ontologie ATM (Air Traffic Management Ontology), développée par la NASA (National Aeronautics and Space Administration) (Keller, 2017b, 2018, 2017a).

2.7.3 Vers un pilote synthétique fondé sur l'architecture cognitive ACT-R

Au regard de l'analyse menée précédemment, l'architecture ACT-R s'est imposée comme le choix le plus pertinent pour concevoir notre pilote synthétique cognitif. Ce choix repose sur la capacité d'ACT-R à modéliser les processus cognitifs humains de manière modulaire et hiérarchique, permettant ainsi une représentation réaliste des mécanismes impliqués dans la prise de décision et l'exécution des tâches en environnement dynamique.

En complément, nous prévoyons d'intégrer l'ontologie de référence pour structurer et formaliser les connaissances expertes du domaine du pilotage. Cette combinaison entre ACT-R et l'ontologie permettra au pilote synthétique de raisonner sur les procédures normalisées, d'interpréter les situations de vol et de fournir une assistance cognitive aux pilotes humains dans le cockpit. Il jouera ainsi le rôle d'un coach intelligent

capable d'anticiper, de diagnostiquer et de corriger les écarts de performance.

Le pilote synthétique que nous implémenterons sera un agent cognitif situé, interagissant en temps réel avec son environnement de travail : le cockpit. Son cycle cognitif suivra trois étapes essentielles :

- Perception de l'environnement via les entrées sensorielles du système ;
- Traitement des informations perçues en s'appuyant sur les connaissances et la mémoire du système ;
- Action sur l'environnement, que ce soit par la génération d'alertes, l'exécution d'une tâche de pilotage ou la suggestion d'actions spécifiques.

Pour réaliser ces opérations, l'agent cognitif s'appuiera sur plusieurs modules de mémoire interconnectés, qui échangeront des informations à travers des tampons (buffers), assurant ainsi un flux de traitement continu et efficace.

En nous basant sur la description du fonctionnement d'un agent ACT-R faite dans (Raluca, 2013), nous détaillons ici les différentes phases de son cycle cognitif :

- La mémoire déclarative contiendra et stockera la connaissance experte en matière de pilotage d'un aéronef, selon les standards du domaine et les connaissances sur l'environnement d'exécution de la tâche. Cela inclura les procédures normales et d'urgence.
- La mémoire procédurale modélisera des productions (condition-action-états) qui s'appliqueront en fonction de l'état de la tâche de pilotage en cours ou de l'environnement de l'aéronef. Ces productions représenteront les savoir-faire du pilote, notamment les séquences d'actions à effectuer lors des phases spécifiques du vol ou en cas d'anomalies.
- Les modules perceptifs et moteurs modéliseront les processus sensoriels de base et assureront l'interaction avec l'environnement.

Ainsi, la modélisation des activités de l'agent cognitif fondé sur ACT-R permettra de représenter :

- Le stockage et la mise à jour des informations sur l'environnement de l'aéronef (conditions météorologiques, trafic aérien, état des systèmes) ;
- Le stockage et la mise à jour des informations sur la tâche de pilotage en cours (phase de vol, objectifs immédiats, paramètres cibles) ;
- Le stockage et la mise à jour des informations sur le pilote (état cognitif, affectif et comportemental), permettant d'adapter l'assistance en fonction de son état ;

— Les procédures de réaction en cas d'alerte ou de déviation par rapport aux normes.

Afin de mener à bien sa mission, l'agent cognitif ACT-R devra être flexible, adaptatif et conscient du contexte opérationnel de la tâche. Il devra non seulement connaître les critères d'une performance optimale ou normative, mais aussi les moyens alternatifs pour atteindre l'objectif. En cas de détérioration des performances ou d'inattention du pilote entraînant une alerte manquée, le pilote synthétique cognitif devra ajuster sa fonctionnalité et adapter sa représentation du pilote. Oliver *et al.* (2019) qualifient cette approche de soutien adaptatif en cas de besoin.

Une caractéristique essentielle du pilote synthétique sera sa capacité d'apprentissage. Grâce aux mécanismes d'apprentissage d'ACT-R, notamment le renforcement des règles procédurales et l'ajustement des paramètres d'activation de la mémoire déclarative, le pilote synthétique pourra améliorer ses performances au fil du temps, permettant ainsi une adaptation progressive aux spécificités de chaque équipage et type d'aéronef.

De plus, le pilote synthétique pourra intégrer un modèle de charge cognitive capable d'évaluer la charge mentale du pilote humain et d'adapter son niveau d'assistance en conséquence. Dans les situations de faible charge, l'assistance sera minimale, favorisant ainsi l'autonomie du pilote. En revanche, lors de phases critiques ou complexes du vol, ou lorsque des signes de fatigue ou de stress seront détectés, le niveau d'assistance augmentera proportionnellement.

La Figure 2.3 présente l'architecture de haut niveau du pilote synthétique, illustrant les principales composantes et leurs interactions.

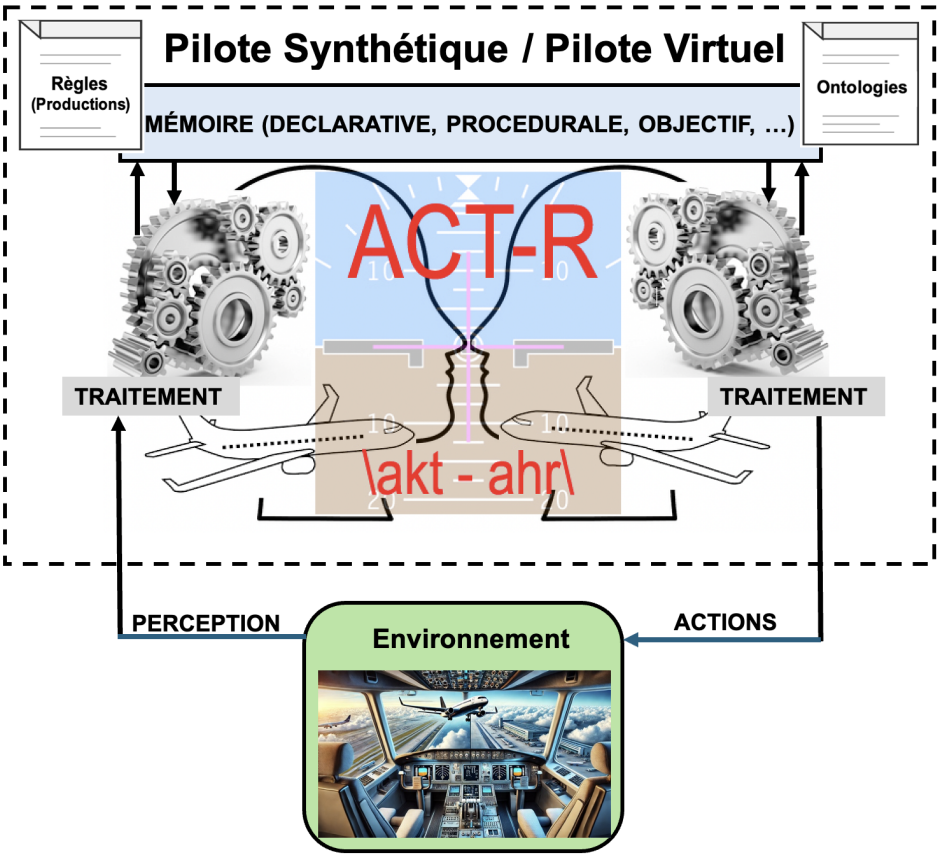


Figure 2.3 – Architecture de haut niveau du pilote synthétique.

2.8 Conclusion

Dans ce chapitre, nous avons exploré les fondements théoriques nécessaires au développement d'un pilote synthétique fondé sur une architecture cognitive et intégrant une ontologie des procédures de pilotage d'un avion. Nous avons distingué le pilote automatique, basé sur des algorithmes déterministes exécutant des tâches prédéfinies, du pilote synthétique, conçu pour imiter les processus cognitifs humains et améliorer la prise de décision en situation de vol.

De plus, l'analyse des ontologies appliquées à l'aviation a mis en évidence l'importance de structurer les connaissances relatives aux tâches de pilotage et à l'environnement opérationnel. Nous avons justifié le choix du modèle de référence ontologique (que nous avons désigné par « ontologie de référence »), qui permet d'assurer une représentation cohérente des processus décisionnels des pilotes humains et des contraintes contextuelles liées à l'environnement de l'aéronef.

Sur cette base, nous proposons de développer un pilote synthétique fondé sur ACT-R, enrichi par une ontologie de référence structurée. Ce système aura pour objectif d'améliorer la prise de décision des pilotes, la gestion des alertes et l'assistance cognitive en vol.

Le pilote synthétique que nous envisageons ne vise pas à remplacer le pilote humain, mais à le comprendre, à l'assister et à améliorer sa performance.

Dans les chapitres suivants, nous décrirons la mise en œuvre concrète de cette approche, en détaillant l'implémentation du modèle cognitif, notamment l'intégration de l'ontologie de référence au sein du pilote synthétique basé sur ACT-R, ainsi que son insertion dans un environnement de simulation.

CHAPITRE 3

INTÉGRATION D'UNE ONTOLOGIE DE RÉFÉRENCE DES PROCÉDURES DE PILOTAGE D'UN AVION DANS UN AGENT COGNITIF FONDÉ SUR L'ARCHITECTURE COGNITIVE ACT-R POUR SIMULER UNE TÂCHE

3.1 Préambule

Ce chapitre présente un pilote synthétique fondé sur l'architecture cognitive ACT-R, dans lequel est intégrée une ontologie de référence des procédures de pilotage. L'objectif est de montrer comment cette intégration est réalisée et comment elle permet à l'agent cognitif de simuler une tâche spécifique du décollage d'un avion de manière fidèle au comportement d'un pilote humain expert. Cette approche vise à reproduire les raisonnements, les décisions et les actions d'un pilote expérimenté, en s'appuyant sur une mémoire structurée et des processus cognitifs inspirés des sciences cognitives (Tamkodjou Tchio *et al.*, 2023).

Nous décrivons en détail le processus de modélisation cognitive mis en œuvre, en insistant sur les mécanismes de perception, de prise de décision et d'action activés par le pilote synthétique à partir de ses connaissances déclaratives et procédurales. Cette démarche démontre que le pilote synthétique peut reproduire les raisonnements et les comportements attendus d'un pilote humain expert, en mobilisant les connaissances formalisées dans l'ontologie de référence et les processus cognitifs de l'architecture ACT-R.

L'agent cognitif développé exécute des tâches de pilotage en suivant le cycle cognitif illustré dans la Figure 3.1, inspiré des procédures de référence utilisées par les pilotes experts. Chaque tâche est décrite selon un ensemble de contraintes, sa supertâche, ses préconditions, un responsable d'exécution et une série d'actions associées.

Ce chapitre s'articule autour de plusieurs sections visant à détailler l'intégration de l'ontologie de référence dans l'architecture cognitive ACT-R pour la simulation d'une tâche de pilotage.

Il débute par une présentation de l'ontologie de référence utilisée pour formaliser les connaissances liées aux procédures de pilotage. Ce modèle de référence, composé d'une ontologie de domaine et d'une ontologie des tâches, permet de structurer les informations relatives à l'environnement d'exécution et aux procédures de pilotage. Les Figures 3.2 et 3.3 illustrent respectivement ces ontologies, mettant en évidence les relations sémantiques entre les concepts.

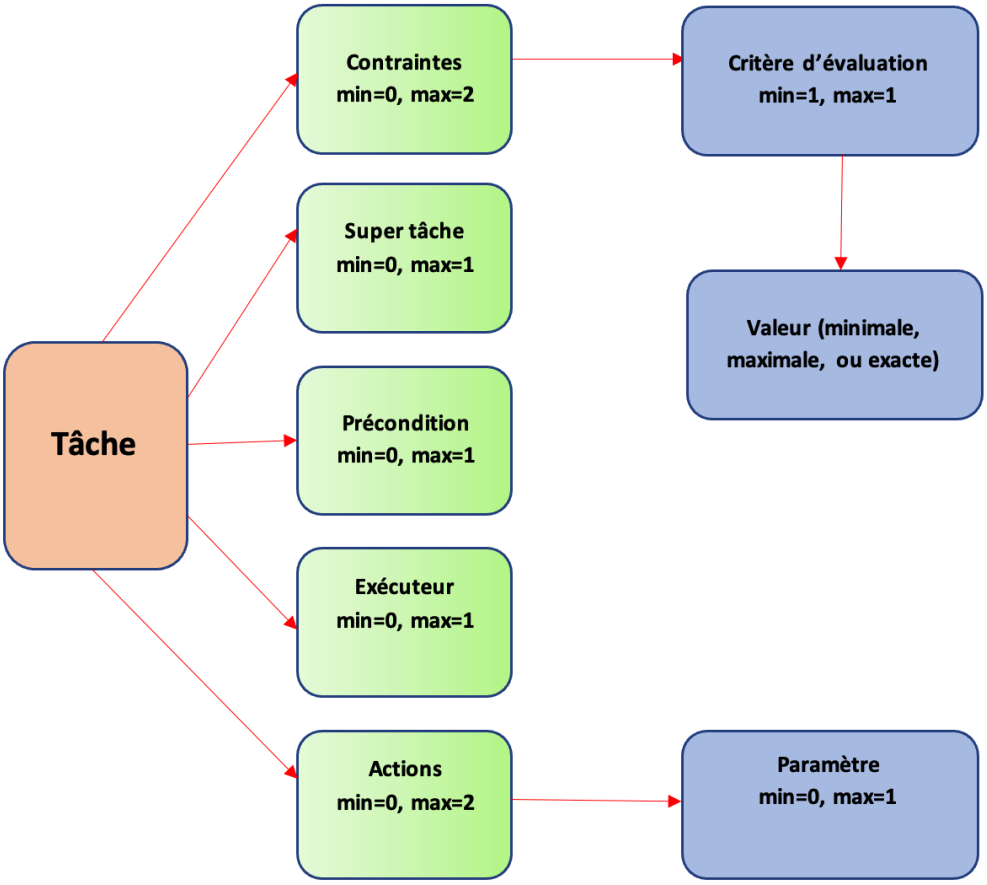


Figure 3.1 – Cycle d'exécution d'une tâche.

Ensuite, le chapitre aborde la modélisation de l'agent cognitif basé sur l'architecture ACT-R. Les composants clés de l'agent sont décrits, notamment sa mémoire déclarative (qui stocke l'ontologie de référence), sa mémoire procédurale (qui contient les règles de production), ainsi que ses modules sensoriels et moteurs (qui lui permettent d'interagir avec son environnement). La Figure 3.4 illustre l'architecture opérationnelle de l'agent, en montrant comment ces différents modules interagissent pour exécuter une tâche.

Le chapitre se concentre ensuite sur la gestion et l'exécution d'une tâche de pilotage par le pilote synthétique. En prenant le décollage comme cas d'étude, il est expliqué comment l'agent identifie et respecte les préconditions, applique les contraintes associées à la tâche et utilise les règles de production pour déduire les actions à entreprendre, tout en mettant à jour son état cognitif. À travers un diagramme de séquence (Figure 3.6), les interactions entre l'agent et l'ontologie de référence lors de l'exécution d'une tâche sont illustrées, ainsi que la manière dont l'agent mobilise ses connaissances pour reproduire les décisions attendues d'un pilote humain.

Par la suite, les résultats expérimentaux et la validation du modèle sont exposés. L'implémentation technique de l'agent à l'aide des bibliothèques pyactr et owlready2 est détaillée, de même que les expériences menées pour valider ses performances. Les résultats obtenus montrent que l'agent est capable de simuler une tâche impliquée dans le décollage, conformément aux procédures expertes du domaine de l'aviation. La Figure 3.8 illustre un exemple de cycle d'exécution d'une tâche, tandis que la Figure 3.9 montre le processus de déploiement et de validation dans un environnement de simulation comme X-Plane.

Enfin, le chapitre se conclut par une synthèse des contributions et une présentation des travaux futurs. Des pistes sont proposées pour améliorer et étendre les capacités de l'agent, notamment en passant de la gestion d'une tâche spécifique à celle d'une procédure complète de décollage. La conclusion du chapitre met en avant les principales contributions de cette étude tout en soulignant l'importance de cette recherche pour le développement de systèmes d'assistance cognitive dans le domaine de l'aviation.

3.2 Integrating an Ontological Reference Model of Piloting Procedures in ACT-R Cognitive Architecture to Simulate Piloting Tasks

Cette section est consacrée à l'article intitulé *« Integrating an Ontological Reference Model of Piloting Procedures in ACT-R Cognitive Architecture to Simulate Piloting Tasks »*, publié par *Guy Carlos Tamkodjou Tchio*, *Marc-Antoine Courtemanche*, *Ange Adrienne Nyamen Tato*, *Roger Nkambou* et *Valéry Psyché* dans l'ouvrage *« Augmented Intelligence and Intelligent Tutoring Systems (ITS 2023) »*, dans le cadre de la *19th International Conference ITS 2023*, tenue à Corfu, en Grèce, du 2 au 5 juin 2023.

Abstract. To accurately replicate the procedures and actions of piloting an aircraft, it is important to create an intelligent system capable of analyzing and executing tasks using established protocols in the field. In this study, we introduce a cognitive agent based on the ACT-R cognitive architecture that incorporates an ontological reference model into its declarative memory. The purpose of this is to simulate the performance of critical piloting tasks, such as takeoff, in a manner similar to that of a human pilot. The agent accomplishes this by utilizing production rules stored in its procedural memory to deduce knowledge captured and formalized by the ontological reference model stored in its declarative memory. Our findings suggest that this approach is a key step towards developing a cognitive agent that can be tested in a real flight simulator, providing insights into how human pilots function in terms of their cognitive and affective behavior.

Keywords : ontology · OWL · domain ontology · task ontology · reference model · cognitive architecture · ACT-R · Pyactr · cognitive agent

1 Introduction

The resolution of critical situations in an aircraft piloting scenario always depends on the pilots, who may need to assume manual control of the aircraft if necessary (Oliver *et al.*, 2019). However, piloting an aircraft is a complex task, and pilots can benefit from the support of a reliable artificial system (Insaurralde et Blasch, 2019). Despite this, pilots may experience lapses in attention, leading to reduced performance in the cockpit. The causes of inattention can be diverse and may depend on factors such as perceptual, attentional, and emotional issues (Dehais *et al.*, 2019). Given the decrease in attention that may occur in the cockpit, it is important for pilots to receive cognitive assistance (Estes *et al.*, 2016). A cognitive model that can track an individual user's operational context and cognitive dynamics can serve as the basis for cognitive assistance during operations (Zhang *et al.*, 2018). Our proposed cognitive model aims to demonstrate the functioning of a human pilot by taking into account the pilot's cognitive, affective, and behavioral dimensions. It should also demonstrate the cognitive cycle of a piloting task while considering the pilot's cognitive, affective, and behavioral states. Currently, our cognitive model simulates the execution of a piloting task based on reference procedures described through an ontology.

To formally represent knowledge related to the pilot's task, task environment, and cognitive processes, ontologies can be used. We have adopted the ontology defined by Courtemanche *et al.* (2022), which serves as a reference model for piloting tasks and models complex procedural knowledge related to aircraft piloting procedures. The ontology allows for the automatic manipulation of knowledge by detailing the necessary information related to the execution environment for each task. It consists of a domain ontology and a task ontology. The domain ontology is specific to the execution environment of the complex task of piloting an aircraft, while the task ontology is the central element of the reference model, linking the different execution parameters. The resolution rules are closely linked to the execution environment, and the decomposition of the task execution with the domain theory is formalized by semantic links between the two ontologies, which supports the execution.

Based on Newell's criteria (Newell, 1990) and Sun's desiderata (Sun, 2004), which define various capacities, properties, and evaluation criteria of cognitive architectures, ACT-R has shown the best characteristics in

modeling complex cognitive tasks. Therefore, we have decided to instantiate it in aviation to simulate a human pilot's mind. ACT-R (Anderson *et al.*, 2004) is a comprehensive, science-based cognitive architecture that has produced models representing the processes involved in driving a car, controlling a single-engine aircraft, allocating visual attention in a cockpit, using and acquiring skills for the flight management system, and modeling cognitive assistance in a cockpit (Oliver *et al.*, 2019). This cognitive architecture integrates the major cognitive functions of memory, learning, perception, and actions, as well as their limits. ACT-R has a production system composed of declarative memory and procedural memory. Declarative memory contains facts such as « Ottawa is the capital of Canada », « Cameroon is a country in Africa », or « 8+2=10 », while procedural memory contains production rules that represent knowledge about how we do things. Examples of such knowledge include how to play chess, drive a car, or perform an arithmetic operation.

This paper aims to build a cognitive agent capable of performing piloting tasks using an ontological reference model containing expert knowledge. To represent the knowledge from the information related to the task, the task environment, and the cognitive processes of the pilot formally, we use the ontology defined by Courtemanche *et al.* (2022), which models complex procedural knowledge related to aircraft piloting procedures. The ontological reference model is stored in the declarative memory of our cognitive agent, and the written production rules that allow for inferences on this knowledge are stored in its procedural memory. Our cognitive agent simulates the execution of piloting tasks by following the current procedures, using the production rules contained in its procedural memory to infer the knowledge captured and formalized by the ontological reference model present in its declarative memory. The cognitive agent can track an individual user's operational context and cognitive dynamics, taking into account their cognitive, affective, and behavioral dimensions. Our ultimate goal is to test the cognitive agent in a real flight simulator to demonstrate and explain the functioning of a human pilot in a piloting context.

Our current work is part of a larger project aimed at developing a cognitive agent capable of performing complex piloting tasks in a flight simulator. The agent will be based on information related to the state of the human pilot (including cognitive and emotional factors), the task being performed by the pilot, and the aircraft environment.

2 The Ontological Reference Model

The reference model formalizes knowledge of piloting procedures found in the technical documentation for executing a flight, both under normal and abnormal circumstances. It aims to provide a standardized frame-

work for comparing pilot execution. The model uses Web Ontology Language (OWL) (W3C, 2012) to capture the complex logical relationships between concepts. The reference model consists of two interrelated ontologies : the domain ontology, which contains knowledge about the execution environment, including inside parameters, outside components, and aircraft systems, and the general task taxonomy specific to aviation, which decomposes tasks and anchors them semantically to the domain ontology. The tasks are decomposed and semantically anchored to the domain ontology making the execution framework highly related to the environment. More details about the reference model are available in Courtemanche (Courtemanche *et al.*, 2022). This reference model contains the data required for feeding the declarative memory of our cognitive agent. The reference of execution in OWL is manipulated to identify the right task to accomplish. In addition to providing a reference for selecting the right task to accomplish, the reference model is used for structuring the execution environment. The domain ontology is a taxonomy list of the environment's parameters and contains the dynamic data associated with the simulation. The cognitive agent uses this ontology to gather important data about the simulation and push new data to the environment. By using this standardized taxonomy about the environment, the reference of execution is semantically related to the domain of execution and is also providing an efficient way for making the data of the simulation environment more accessible.

The class diagram depicted in Figure 3.2 represents the domain ontology, while Figure 3.3 displays the task ontology with an attentional layer (Ghaderi *et al.*, 2022b). For the purpose of our work, we only consider the classes represented by the blue squares in the task ontology. However, the cognitive agent may utilize the attention class in the future to enhance its functionality.

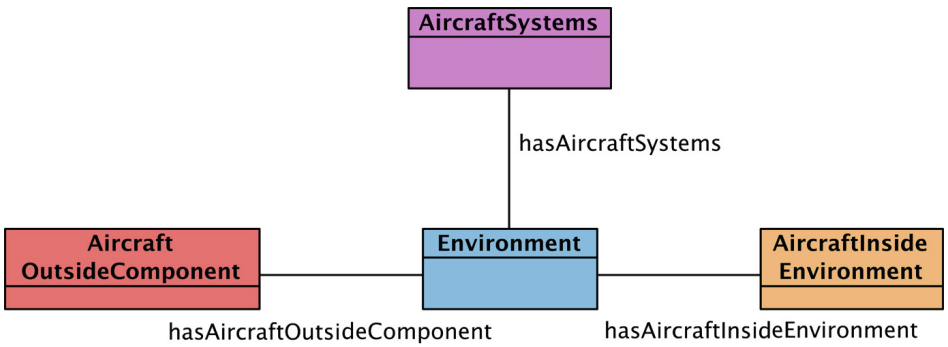


Figure 3.2 – Domain ontology excerpt.

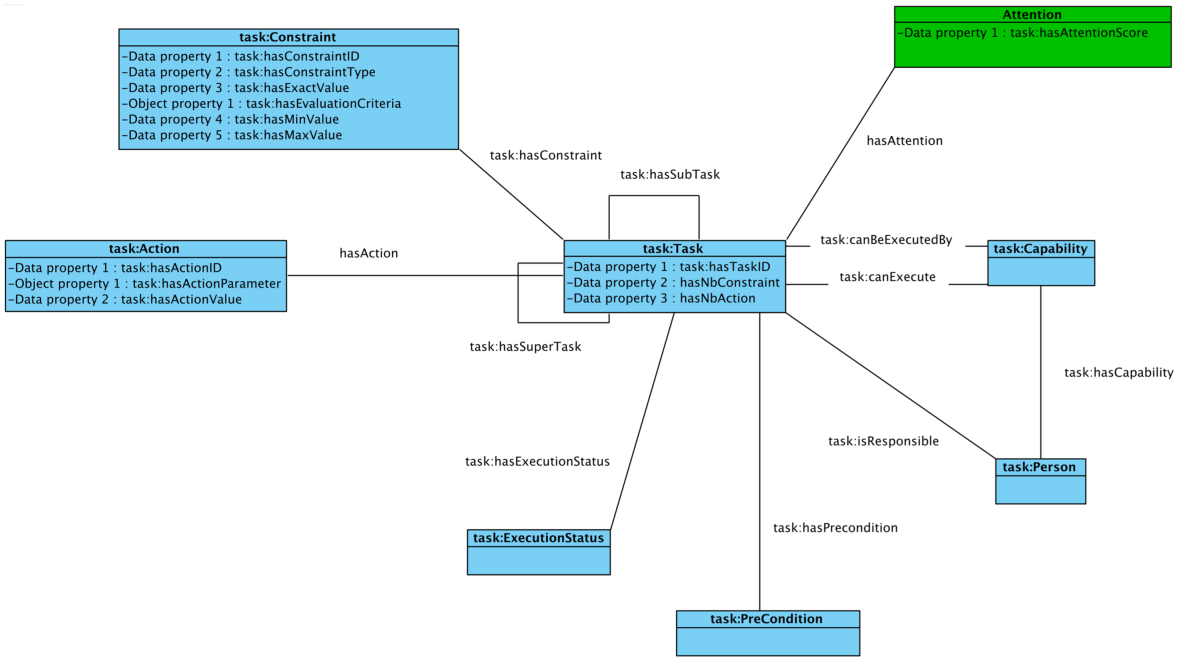


Figure 3.3 – Task ontology with an attentional layer.

3 ACT-R Cognitive Agent Modeling for Piloting Tasks

The ACT family of architectures, developed by Anderson and his colleagues, is the oldest (since 1973) and best known, with the highest number of scientific articles devoted to it (Kotseruba et Tsotsos, 2020). The aim of ACT-R (Adaptive Control of Thought - Rational) is to model the human mind (Anderson *et al.*, 2004), exploring the four dimensions of artificial intelligence : thinking like a human, thinking rationally, acting like a human, and acting rationally (Russell et Norvig, 2020). ACT-R is a production system consisting of a declarative memory that contains facts and a procedural memory that contains production rules. The declarative module recognizes what is presented to the model and calculates the activation of rules, while the procedural module calculates the utility of each activated rule and triggers the most appropriate one. Cognition emerges from the interaction between procedural and declarative structures.

Based on Anderson's cognitive architecture (Anderson *et al.*, 2004), we propose an architecture for our cognitive agent as shown in Figure 3.4. The proposed cognitive agent is situated in the cockpit or an aircraft flight simulator. To interact with its environment, the agent has three main components : the sensory module, the pattern matcher, and the motor module. The sensory module enables the agent to perceive relevant information from its environment, such as the status of the aircraft, the position of the controls,

and the actions of the human pilot. The pattern matcher allows the agent to recognize and process the perceived information using its declarative and procedural memories. Finally, the motor module enables the agent to act on the environment by sending commands to the simulator or the aircraft systems. Currently, the cockpit is simulated using the ACT-R runtime environment, which provides a platform for testing and evaluating the cognitive agent's performance in realistic scenarios.

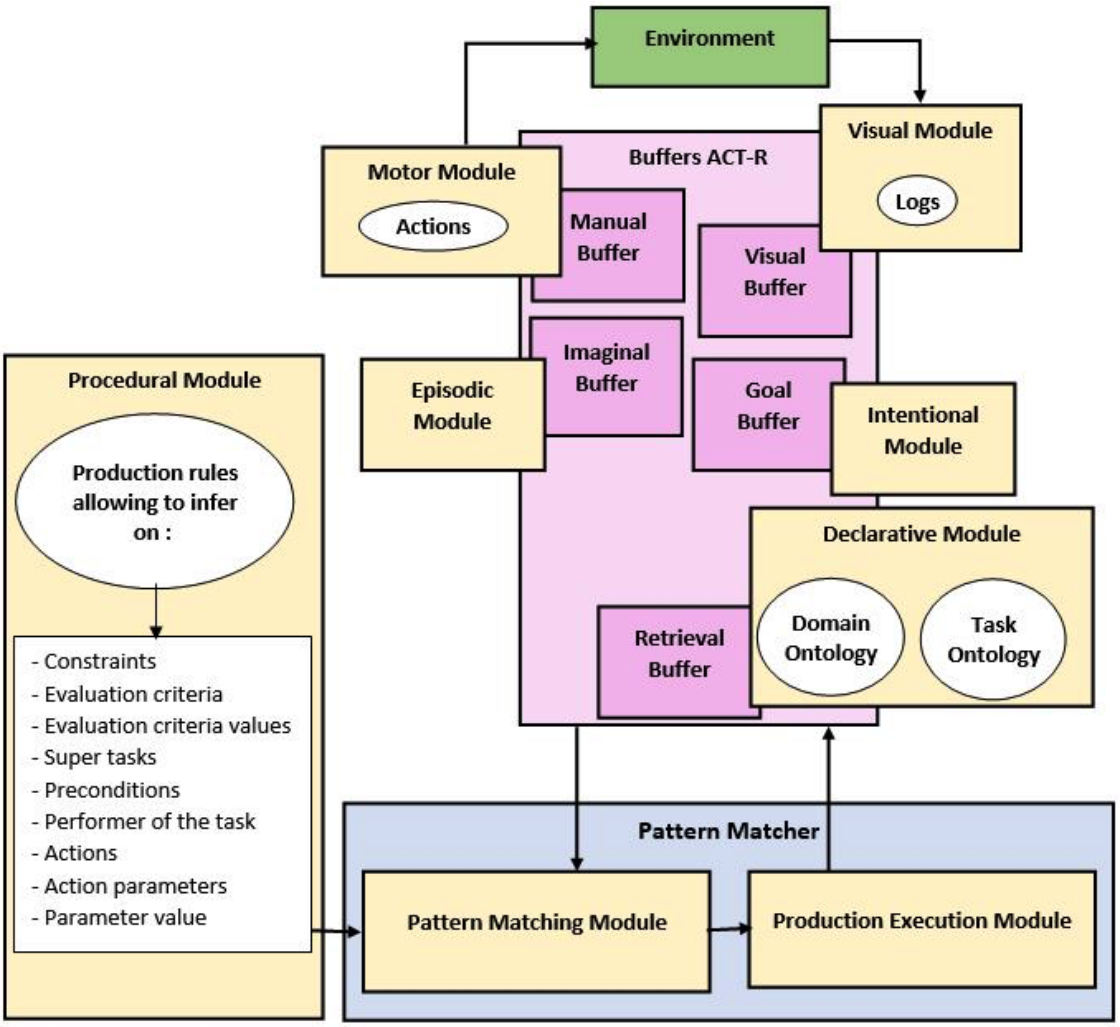


Figure 3.4 – Cognitive agent operating architecture.

The cognitive agent has several buffers working together, which are essential components of cognition (Atkinson et Shifrin, 1968). Memory will be an indispensable part of our agent, allowing it to perform various functions, including but not limited to : Storing intermediate results of calculations, Learning, and Adapting to a changing environment.

The memory of our agent is divided into short-term and long-term memory. Short-term memory is composed of sensory and working memory. The sensory memory stores and preprocesses recent perceptions, as well as actions to be performed. The Visual Buffer is related to sensory memory and contains logs captured from the environment, biases introduced relating to deviations, incapacities, and pilot behavior. The Manual buffer contains the tasks, constraints, and actions to be performed on the environment after the processing performed by the pattern matcher. The working memory contains items related to the task being executed. The Imaginal Buffer maintains an internal picture of the information associated with the current cognitive process, providing contextual information relevant to the current task. The Goal Buffer contains the goal to be achieved.

Long-term memory, on the other hand, is divided into procedural memory and declarative memory. Procedural memory stores knowledge about actions to take under certain conditions. In our agent, procedural knowledge is represented by a set of "if..then" rules. Declarative memory stores factual knowledge, including the ontological reference model. The Buffer Retrieval contains non-declarative, implicit knowledge.

The pattern matcher is a central module in the cognition of our cognitive agent, and it includes two modules : the production rule selection module and the production rule execution module. The production rule selection module searches the procedural memory for the production rule corresponding to the buffers' current state. Only one production rule can be executed at any given time. Once executed, this production rule can modify the buffers and change the system's state. Thus, cognition is a series of production rules triggered by our cognitive agent. The production rules are selected to be executed using the utility learning mechanism, which selects the rule with the highest utility. Additionally, the pattern matcher represents and processes information in a hybrid way that is both symbolic and connectionist (Raluca, 2013).

4 Management and Execution of Tasks by the Cognitive Agent

From the class diagram (Figure 3.3), it appears that executing a task by the cognitive agent involves determining its constraints, its super task, its precondition, its execution manager, and its actions. Figure 3.5 below shows the information related to takeoff in the Protege ontology editor. This information is composed of task and domain ontology values.

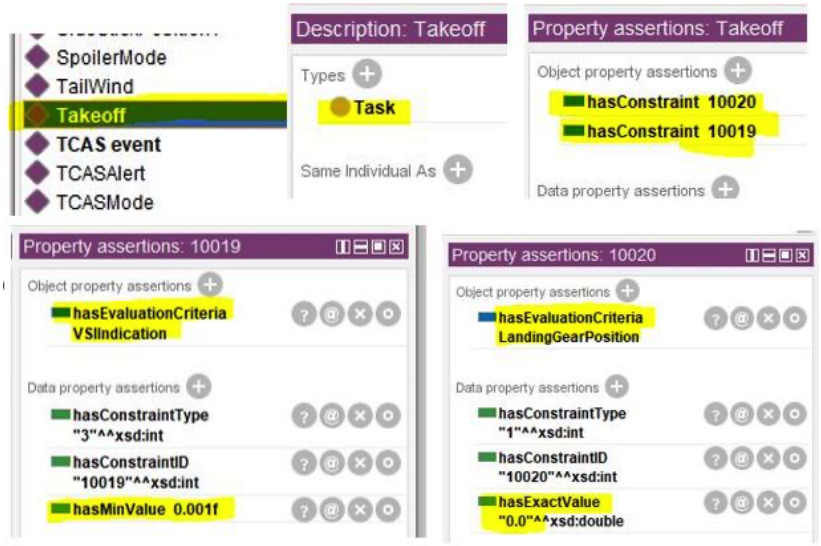


Figure 3.5 – Information related to the Takeoff task.

In UML (Unified Modeling Language) formalism, a sequence diagram is an interaction diagram that expresses dynamically how different objects collaborate. In Figure 3.6, we use this diagram to present the functioning of our cognitive agent in executing a task and its interactions with the reference model.

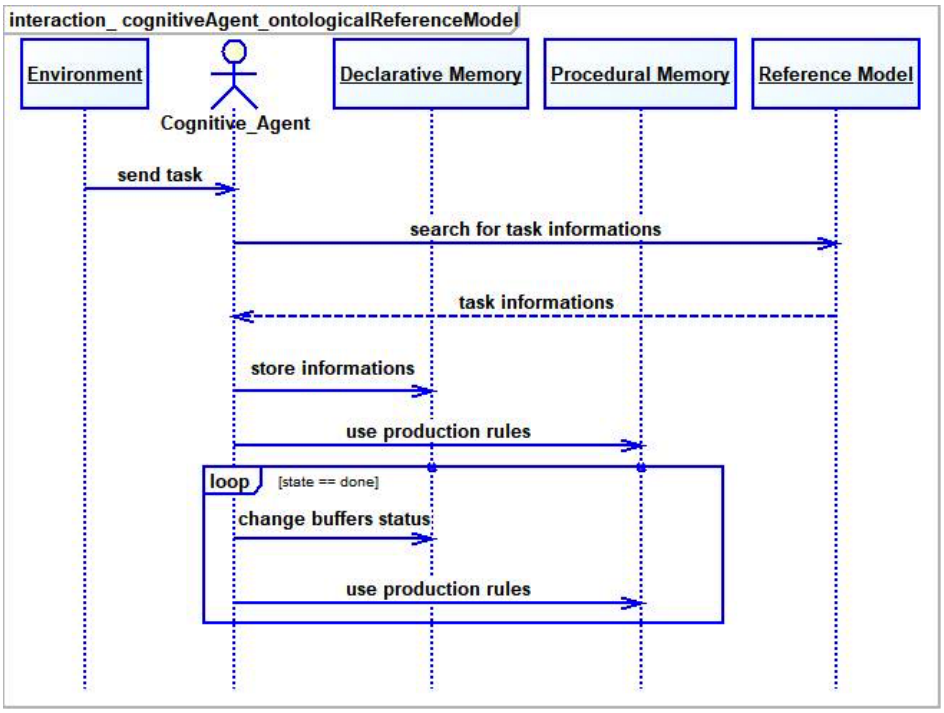


Figure 3.6 – Sequence diagram of the execution of a task by the cognitive agent.

5 Methodology, Results and Model Validation

5.1 Methodology

The ACT-R theory has an official implementation in LISP and has subsequently been implemented in several programming languages, including Java (jACT-R and Java ACT-R), Swift (PRIM), Python2 (ccm), and Python3 (pyactr) (Brasoveanu et Dotlačil, 2020). The latest implementation, pyactr, is very close to the official Lisp implementation and was developed to facilitate the transfer of ACT-R modeling skills between Python and Lisp, and due to the popularity of the Python programming language. Pyactr is a Python library used to create and run cognitive models based on ACT-R, while Owlready2 is a Python module for ontology-oriented programming that allows loading OWL 2.0 ontologies and manipulating them transparently in Python.

We implemented our cognitive agent using the pyactr library and loaded the ontological reference model into the declarative memory of our agent using owlready2 to infer and perform piloting tasks. To perform a task, our cognitive agent retrieves it and searches the ontology for related information, which it places in its declarative memory. It also places the objective to be achieved in its goal buffer and uses its imaginal buffer to maintain an internal image of the information associated with the task being performed. The agent places the production rules in its procedural memory and uses the pattern matcher, a central module in the cognition of our cognitive agent, to choose the production rule corresponding to the current state of the buffers to execute. Once executed, the system's state changes and the mechanism starts again until the goal buffer is in the desired state. Figure 3.7 presents the knowledge extraction in the ontological reference model by the ACT-R cognitive agent in a simple way.

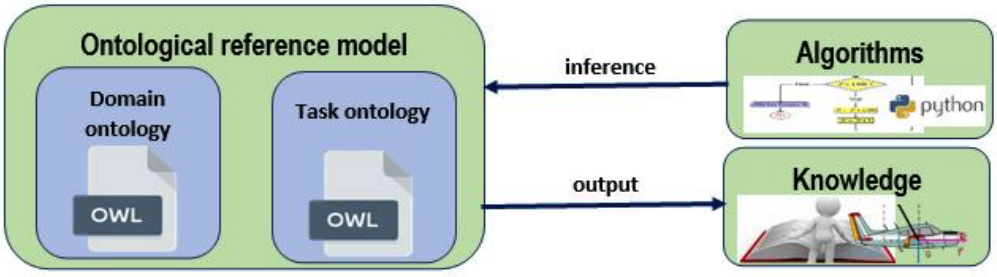


Figure 3.7 – Inference made by ACT-R cognitive agent.

The algorithm 1 gives details on how the cognitive agent executes a task.

Algorithm 1 : Algorithm for executing a task involved in takeoff.

Input : ReferenceModel

Parameter : task

Output : executionTime, preconditions, performer, superTask, actions, parameters, constraints, evaluationCriterias

```
receive(task)
infos = search_information_in_referenceModel(task)
pattern_matcher(infos.superTask)
print(executionTime(infos.superTask))
pattern_matcher(infos.preconditions)
print(executionTime(infos.preconditions))
pattern_matcher(infos.performer)
print(executionTime(infos.performer))
if len(infos.constraints) > 0 then
    pattern_matcher(infos.constraints)
    print(executionTime(infos.constraints))
    print(evaluationCriterias)
end if
if len(infos.actions) > 0 then
    pattern_matcher(infos.actions)
    print(executionTime(infos.actions))
    print(parameters)
end if
buffers = [infos.executionTime, infos.superTask, infos.preconditions, infos.performer, infos.constraints,
infos.actions, infos.evaluationCriterias, infos.parameters]
return buffers
```

The functions and parameters used in the algorithm have the following meaning :

- **infos** is a variable that contains for the current task : the super task, the precondition, the performer, constraints and actions.
- **receive()** is a function that takes a task as a parameter and saves it in the declarative memory of our

cognitive agent

— **pattern_matcher(X)** searches in the procedural memory of the cognitive agent the rules allowing to execute the parameter X of the current task.

— **search_information_in_referenceModel(X)** places in the **Infos** variable the information about : super task, precondition, performer, constraints, and actions.

— **print(executionTime(X))** displays the execution times of X.

5.2 Results and Model Validation

The experiments carried out show that the model implemented successfully executed a piloting task by demonstrating the cognitive cycle of solving the task, as shown in the example of Figure 3.8, which represents a part of the complex task of takeoff.

Figure 3.8 – Task cognitive cycle.

The obtained results demonstrate that the cognitive agent verifies the constraints and performs the actions in a manner similar to a human expert over time. According to Courtemanche *et al.* (2022), the terminology elements represented by the reference model are the same as those of the X-Plane simulation environment and the A320 aircraft. Therefore, to verify the results provided by our cognitive model in a real aircraft flight environment, we plan to deploy the developed models with the X-Plane flight simulator. However, with some adjustments, it may also be possible to use our cognitive agent in a real flight simulator. As ACT-R is a scientifically based theory for modeling human cognitive performance (Smart *et al.*, 2016), the deployment

process will follow the scheme shown in Figure 3.9.

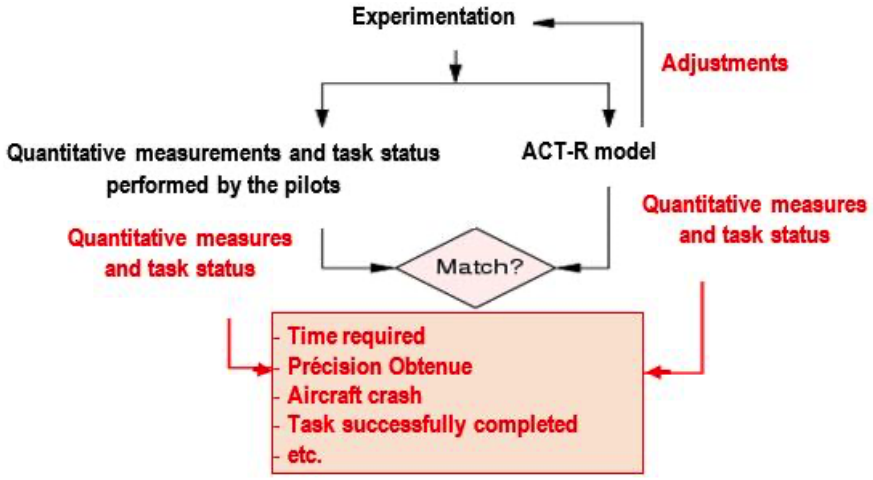


Figure 3.9 – Deployment and tests.

As described in (Raluca, 2013), the validation will consist of collecting the quantitative measurements produced by the ACT-R cognitive agent in the X-plane simulation environment and will compare them to the quantitative measurements made by the pilots, as shown in Figure 3.10. Aviation experts will then be invited to validate the results produced by the cognitive agent in X-plane. Thus validated, the cognitive agent will be used as a pilot simulating a piloting task in a particular context.

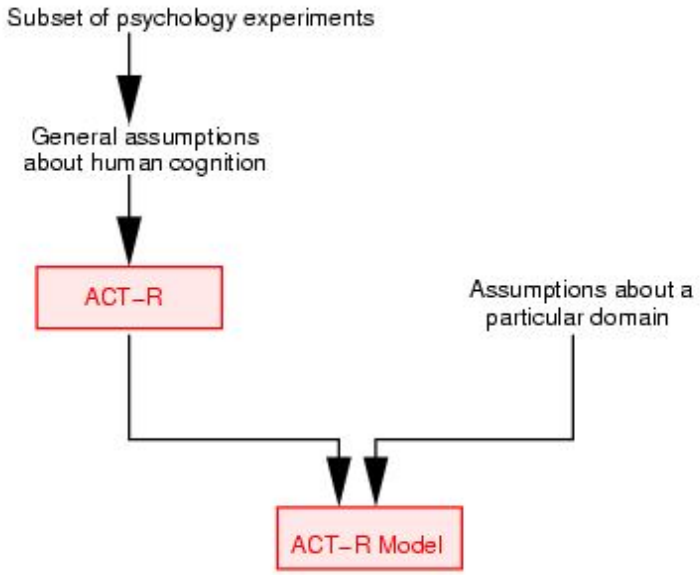


Figure 3.10 – Validation of an ACT-R model (Raluca, 2013).

6 Discussion and Open Problems

The results demonstrate that the cognitive agent can execute and present the cognitive cycle of a complex piloting task, providing insight into the cognitive processes involved in human piloting. Integrating an ontological reference model into the ACT-R cognitive architecture allows the cognitive agent to simulate piloting tasks according to expert procedures, similar to a human pilot. The appropriate rule to execute a piloting task at a given time is selected using utility learning to select the rule with the highest utility. However, as the piloting task is simulated, the ACT-R pattern matcher determines the time to check a constraint or execute an action.

According to Varela *et al*. (1991), cognition is embodied, situated, and social. Embodied cognition is determined by the structure of the organism, with the body being in the brain and not just the brain in the body. Situated cognition means that the environment's state determines cognition. Social cognition means that cognition is a collective cognitive system that emerges from the interaction of individual cognitive systems. One open problem is situating the cognitive agent in an environment like Xplane. Another challenge is using the attentional dimension of piloting tasks presented in the task ontology (Figure 3.3) to execute a piloting task.

7 Conclusion

In this paper, we introduce a cognitive agent based on the ACT-R cognitive architecture that utilizes an ontological reference model to perform tasks in the aviation domain. The declarative memory of the agent captures and formalizes knowledge from the domain ontology, which represents the internal and external environments and aircraft navigation systems, and the task ontology, which captures and formalizes piloting procedures. Using a class diagram, we modeled the complex structure of a task and the relationships between its components. When executing a task, the cognitive agent searches the knowledge formalized by the reference model to perform the task. We presented its behavior and interactions with other components using a sequence diagram. Our simulations demonstrated the cognitive execution cycle of a complex piloting task, and the results showed that the agent executed the task similarly to a human pilot.

As the reference model continues to incorporate the cognitive dimension associated with each task, a second perspective is to develop a version of the cognitive agent that integrates parameters such as the pilot's attention or emotions. Additionally, since cognition is embodied, social, and situated, a third perspective is

to immerse the cognitive agent in a simulator such as Xplane or other flight simulators with suitable adaptations. Finally, a fourth perspective is to use the cognitive agent to manage deviations in a cockpit.

Acknowledgments. We acknowledge the support of CRIAQ, the Natural Sciences and Engineering Research Council of Canada (NSERC), CAE, Bombardier, and BMU.

3.3 Discussion et Conclusion

Dans ce chapitre, nous avons présenté une approche d'intégration de l'ontologie de référence dans un agent cognitif fondé sur l'architecture cognitive ACT-R pour la simulation de tâches de pilotage impliquées dans le décollage d'un avion. Cette approche a permis de structurer et d'exploiter les connaissances expertes en aviation, afin de doter l'agent cognitif de la capacité à exécuter une tâche complexe de manière autonome.

Nous avons démontré que l'utilisation de l'ontologie de référence offre un cadre rigoureux pour organiser et formaliser les connaissances essentielles à la prise de décision et à l'exécution des tâches de pilotage. L'agent cognitif développé, basé sur ACT-R, exploite ces connaissances en mobilisant ses mémoires déclarative et procédurale, en appliquant des règles de production et en suivant un cycle cognitif similaire à celui d'un pilote humain.

Pour l'agent cognitif, exécuter une tâche revient à déterminer ses contraintes, sa supertâche, ses préconditions, son responsable d'exécution et ses actions. Une tâche peut comporter au plus deux contraintes (entre 0 et 2), chacune étant définie par une valeur supérieure ou égale à 10 000. Chaque contrainte est associée à un unique critère d'évaluation, lequel est paramétré par une seule valeur (exacte, minimale ou maximale). Par ailleurs, une tâche peut avoir au plus une supertâche (entre 0 et 1), qui représente sa tâche mère, et au plus une précondition (entre 0 et 1), correspondant à une autre tâche devant être accomplie avant son exécution. Chaque tâche est également liée à au plus un responsable d'exécution (entre 0 et 1), pouvant être soit le « pilot flying », soit le « pilot monitoring ». Enfin, une tâche peut inclure au plus deux actions (entre 0 et 2), chacune étant définie par une valeur comprise entre 1 et 999. Une action peut, quant à elle, être associée à au plus un paramètre d'action (entre 0 et 1), caractérisé par une seule valeur lorsqu'il est présent.

Notre implémentation utilise la bibliothèque pyactr, une implémentation Python proche de l'implémentation officielle en LISP de la théorie ACT-R, permettant de créer et d'exécuter des modèles cognitifs. Nous

avons également utilisé owlready2 pour charger et manipuler les ontologies de manière transparente en Python, facilitant ainsi l'intégration de l'ontologie de référence dans la mémoire déclarative de notre agent cognitif.

L'illustration avec une tâche impliquée dans le décollage a permis de valider le fonctionnement de notre approche, montrant que l'agent est capable d'interpréter les contraintes, d'évaluer les préconditions et d'exécuter les actions requises de manière cohérente avec les procédures standard. Les résultats obtenus confirment la pertinence du modèle proposé et ouvrent des perspectives intéressantes visant à améliorer et à étendre les capacités de l'agent cognitif développé dans ce chapitre

Cependant, plusieurs défis restent à relever pour affiner ce modèle, notamment l'extension des capacités du pilote synthétique à l'exécution de la procédure complète de décollage en situation normale, plutôt qu'à une seule tâche spécifique impliquée dans le décollage d'un aéronef. Cette extension sera abordée dans le chapitre suivant de cette thèse.

CHAPITRE 4

VERS UNE ASSISTANCE COGNITIVE DANS LES TÂCHES DE PILOTAGE D'UN AVION EN SITUATION NORMALE : DÉVELOPPEMENT D'UN PILOTE SYNTHÉTIQUE FONDÉ SUR L'ARCHITECTURE ACT-R

4.1 Préambule

Dans ce chapitre, nous présentons le développement d'un pilote synthétique fondé sur l'architecture ACT-R, s'inscrivant dans une démarche visant à mettre en place une assistance cognitive pour les tâches de pilotage d'un avion en situation normale. Ce pilote synthétique étend les capacités de l'agent cognitif introduit au chapitre précédent en intégrant un modèle de coaching cognitif capable d'assister les pilotes en temps réel lors de l'exécution de la procédure complète de décollage en situation normale. Ce système est conçu pour détecter les écarts par rapport aux procédures standard et formuler des recommandations adaptées, en s'appuyant sur une base de connaissances ontologique (l'ontologie de référence) pour analyser la situation de manière dynamique et contextualisée (Tamkodjou Tchio *et al.*, 2024a).

Le chapitre débute par la présentation des fondements théoriques de l'ontologie de référence et de l'architecture cognitive ACT-R, qui constituent les piliers du pilote synthétique développé.

Il se poursuit avec un état de l'art structuré autour de quatre domaines clés : la représentation ontologique des connaissances en aéronautique, l'exécution automatique des procédures de pilotage basées sur des ontologies, la détection automatique des déviations en vol, ainsi que la modélisation de tâches de pilotage par un agent cognitif ACT-R.

Le chapitre aborde ensuite l'architecture du pilote synthétique cognitif en détaillant ses principaux composants, notamment sa mémoire déclarative (qui intègre l'ontologie de référence), sa mémoire procédurale (qui contient les règles de production et les règles SWRL), et ses modules perceptifs et moteurs. La Figure 4.2 présente l'architecture interne du pilote synthétique, illustrant comment ses différents modules interagissent pour simuler les processus cognitifs impliqués dans le pilotage.

Une section est ensuite consacrée à la méthodologie et aux résultats expérimentaux. L'implémentation technique du pilote synthétique y est décrite à l'aide des bibliothèques pyactr, pour la modélisation ACT-R, et owlready2, pour la manipulation de l'ontologie. Le diagramme de séquence de la Figure 4.3 illustre les

interactions au sein de l'agent lors de l'exécution d'une tâche, tandis que l'Algorithme 1 présente les étapes de la procédure de décollage. Les résultats des simulations sont analysés, démontrant la capacité de l'agent à exécuter des procédures complètes de décollage conformément aux standards et à gérer efficacement les récupérations en cas de déviation. Les Figures 4.6 et 4.7 illustrent respectivement le graphe de tâches généré pour la procédure standard de décollage et les adaptations requises en cas de situations non prévues par l'ontologie de référence. Par ailleurs, la section se penche sur le fonctionnement du pilote synthétique, en précisant les deux modes d'opération possibles, à savoir le mode autonome et le mode interactif, illustrés par la Figure 4.8.

Enfin, le chapitre s'achève sur une synthèse des contributions apportées, suivie d'une discussion sur les perspectives d'évolution du modèle. En particulier, l'extension du pilote synthétique au traitement des procédures anormales pouvant survenir lors de la phase de décollage est envisagée comme une étape clé des travaux futurs.

La métrique utilisée pour valider les résultats produits par le pilote synthétique est la Distance d'Édition de Graphe (Graph Edit Distance — GED) (Serratosa, 2021; Gao *et al.*, 2010). C'est une mesure fondamentale en théorie des graphes, permettant d'évaluer la similarité structurelle entre deux graphes.

La GED correspond au coût minimal d'une séquence d'opérations d'édition élémentaires permettant de transformer un graphe G_1 en un graphe G_2. Les opérations généralement prises en compte sont les suivantes :

- Insertion de nœud ;
- Suppression de nœud ;
- Substitution de nœud ;
- Insertion d'arête ;
- Suppression d'arête ;
- Substitution d'arête.

Mathématiquement, la GED peut être exprimée comme :

$$GED(G_1, G_2) = \min_{(e_1,\ldots,e_k)\in T(G_1,G_2)} \sum_{i=1}^{k} c(e_i) \tag{4.1}$$

Où :

- G_1 et G_2 sont les deux graphes à comparer ;
- $T(G_1, G_2)$ représente l'ensemble de tous les chemins d'édition possibles pour transformer G_1 en G_2 ;
- $e_1, \ldots, e_k$ est une séquence d'opérations d'édition formant un chemin d'édition ;
- $c(e) \geq 0$ est le coût associé à l'opération d'édition e.

Une formulation équivalente de la GED est donnée par la formule suivante :

$$GED(G_1, G_2) = \min_{\gamma \in \Gamma(G_1, G_2)} \sum_{(u,v) \in \gamma} c(u, v) \tag{4.2}$$

Où :

- G_1 et G_2 sont les deux graphes à comparer ;
- $\Gamma(G_1, G_2)$ est l'ensemble des correspondances (ou mappings) possibles entre les nœuds et arêtes de G_1 et G_2 ;
- $c(u, v)$ est le coût de l'opération d'édition pour transformer l'élément u de G_1 en l'élément v de G_2.

La version normalisée de la GED a pour objectif de produire une valeur comprise entre 0 et 1, où 0 indique une similarité parfaite entre les deux graphes. Elle peut s'exprimer comme suit :

$$GED_{\text{norm}}(G_1, G_2) = \frac{GED(G_1, G_2)}{\max\left(|V_{G_1}| + |E_{G_1}|,\ |V_{G_2}| + |E_{G_2}|\right)} \tag{4.3}$$

Où :

- G_1 et G_2 sont les deux graphes à comparer ;
- V_G représente l'ensemble des nœuds du graphe G ;
- E_G représente l'ensemble des arêtes du graphe G ;
- $|V_G|$ représente le nombre de nœuds (vertices) dans le graphe G ;
- $|E_G|$ représente le nombre d'arêtes (edges) dans le graphe G.

Voici comment interpréter les valeurs de la GED_{norm} :

- $GED_{\text{norm}} = 0$: les deux graphes G_1 et G_2 sont identiques. Ils possèdent les mêmes nœuds, les mêmes arêtes, une structure équivalente et, le cas échéant, les mêmes étiquettes (attributs sémantiques) ;
- $GED_{\text{norm}} \approx 0$: les deux graphes sont très similaires, avec seulement quelques différences mineures ;
- $GED_{\text{norm}} \approx 1$: les deux graphes sont structurellement très différents ;
- $GED_{\text{norm}} = 1$: les deux graphes sont complètement dissemblables.

Selon les besoins, il est possible d'adapter la formule normalisée de manière à ce que la valeur 1 représente une similarité parfaite (et non une dissimilarité). Pour cela, il suffit de retrancher la valeur normalisée précédente de 1. La formule devient alors :

$$GED_{\text{norm}}(G_1, G_2) = 1 - \frac{GED(G_1, G_2)}{\max\left(|V_{G_1}| + |E_{G_1}|,\ |V_{G_2}| + |E_{G_2}|\right)} \tag{4.4}$$

Pour valider les résultats obtenus par le pilote synthétique dans ce chapitre, G_1 représente le graphe orienté des tâches générées dynamiquement par le pilote synthétique, tel qu'illustré à la Figure 4.6. En revanche, G_2 correspond au graphe des tâches issues des procédures de référence, définies par des experts et consignées dans le diagramme d'états-transitions de la Figure 4.5.

La GED a permis d'évaluer les aspects suivants :

- **Correspondance des nœuds** : vérification que toutes les tâches (nœuds) du diagramme d'états-transitions de référence sont présentes dans le graphe généré par le pilote synthétique. Par exemple, pour le scénario *Takeoff* (Figure 4.5), nous avons confirmé que les tâches telles que *1001* (tâche initiale) et *1035* (tâche finale) étaient bien incluses dans le graphe généré (Figure 4.6).
- **Cohérence des arcs** : validation du respect des transitions entre les tâches dans le graphe généré, conformément à la logique procédurale définie dans l'ontologie de référence. Cela inclut la vérification des dépendances séquentielles et conditionnelles entre les tâches, telles que les préconditions et contraintes illustrées dans la Figure 4.7 (a, b et c). Ainsi, une transition produite est considérée comme valide si, et seulement si, elle respecte les contraintes logiques et les préconditions définies

dans l'ontologie de référence.

Le tableau ci-dessous présente le coût associé à chaque opération d'édition :

Opération	**Coût**
Suppression d'un nœud	1
Insertion d'un nœud	1
Substitution de nœud	1
Suppression d'une arête	1
Insertion d'une arête	1
Substitution d'arête	1

Table 4.1 – Coût associé à chaque opération d'édition dans le calcul de la GED.

Comme les graphes G_1 et G_2 sont identiques dans ce cas, aucune opération d'édition n'est nécessaire. On a donc :

$$GED(G_1, G_2) = 0$$

Le calcul de la GED normalisée s'effectue comme suit :

$$GED_{\text{norm}}(G_1, G_2) = \frac{GED(G_1, G_2)}{\max\left(|V_{G_1}| + |E_{G_1}|,\ |V_{G_2}| + |E_{G_2}|\right)}$$

Dans notre cas, comme $GED(G_1, G_2) = 0$, on obtient :

$$GED_{\text{norm}}(G_1, G_2) = \frac{0}{\max\left(|V_{G_1}| + |E_{G_1}|,\ |V_{G_2}| + |E_{G_2}|\right)} = 0$$

Pour le scénario « Takeoff », le graphe généré par le pilote synthétique est structurellement et sémantique-

ment identique au graphe de référence issu des procédures standards.

Par conséquent, la distance d'édition de graphes calculée entre G_1 et G_2 est nulle, soit :

$$GED(G_1, G_2) = 0 \quad \text{et} \quad GED_{\text{norm}}(G_1, G_2) = 0$$

Ce résultat indique une correspondance parfaite entre les deux graphes, sans qu'aucune opération d'édition ne soit nécessaire.

Ainsi, les actions produites par le pilote synthétique peuvent être considérées comme pleinement conformes aux standards définis, ce qui valide la fidélité du modèle généré pour ce scénario.

4.2 Towards Cognitive Coaching in Aircraft Piloting Tasks : Building an ACT-R Synthetic Pilot Integrating an Ontological Reference Model to Assist the Pilot and Manage Deviations

Cette section est dédiée à la présentation de l'article *« Towards Cognitive Coaching in Aircraft Piloting Tasks : Building an ACT-R Synthetic Pilot Integrating an Ontological Reference Model to Assist the Pilot and Manage Deviations »*, rédigé par *Guy Carlos Tamkodjou Tchio*, *Roger Nkambou*, *Ange Adrienne Nyamen Tato* et *Valéry Psyché*, et publié dans l'ouvrage *« Generative Intelligence and Intelligent Tutoring Systems (ITS 2024) »*. Cet article a été présenté lors de la *20th International Conference on Intelligent Tutoring Systems (ITS 2024)*, qui s'est tenue à Thessalonique, en Grèce, du 10 au 13 juin 2024.

Abstract. Most aviation assistance systems do not take into account the pilot's actual cognitive state when providing assistance. Yet, especially in critical situations such as aircraft takeoff, it is important to determine whether the information presented has been correctly processed and understood by the pilot, or whether some has been omitted or misinterpreted. This paper presents a cognitive synthetic pilot based on the ACT-R cognitive architecture and integrating a reference ontology of standard piloting procedures as a knowledge base. The main purpose is to serve as a coaching system for training pilots in simulation environments to perform critical piloting tasks such as takeoff, by exploiting the advantages of a rich semantic representation of the aeronautical context. In this way, the ontology formally models the expert piloting procedures that will be used by the synthetic pilot to advise an trainee pilots during training sessions.

For this, we use two types of ontologies to model the pilot's work : a task ontology describing all the actions the pilot must perform, and a domain ontology containing knowledge about the execution environment. The synthetic pilot uses semantic rules and a reasoner for task automation. The rules define when a task can be executed. The reasoner analyzes these rules and the context to decide which actions to execute. Over time, the actions to be executed are presented in the form of a complex dynamic 3D graph, thus allowing better visualization of the tasks to be performed and intelligent automation of the flight procedures described in the ontological reference model to assist the pilot. This work is an intermediate step towards the implementation of a complete cognitive assistance for novice pilots in a simulation environment. The ultimate goal is to extend the capabilities of the synthetic pilot through machine learning, by analyzing real flight data to extract typical pilot behavioral profiles.

Keywords : ACT-R Cognitive Architecture · Ontology · Graph Theory · Cognitive Agent · Synthetic Pilot · SWRL Rules · OWL · Pyactr

1 Introduction

In daily life, the suggestions of our peers significantly influence many of our decisions, whether it's choosing a scientific journal, a candidate for recruitment or a movie (Mahmood et Ricci, 2009; McSherry et Mironov, 2009). Moreover, with the rise of online platforms and their recommendation algorithms, the impact of peer reviews now extends beyond one's close social circle. Technologies enable us to receive personalized proposals based on the analysis of our past preferences and those of a large panel of users with similar tastes. Thus, the tendency to rely on the advice and recommendations from others, whether from those around us or from automated systems, is increasingly shaping our choices in various domains. This tendency to rely on recommendations and feedback is also present in more technical fields like flight training, where new pilots rely on the advice and debriefings of their instructors, who act as guides and experienced peers (Bouzekri *et al.*, 2019). Additionally, to enhance the relevance of suggestions to users, recommendation algorithms increasingly incorporate contextual information about the user's situation. These contextual data, aiding in refining recommendations, can either be provided directly by the user or collected automatically by sensors and other devices capable of determining the user's current situation (Adomavicius et Tuzhilin, 2015). Despite the availability of automated systems dedicated to pilot training, pilots often face issues of inattention, leading to decreased performance during flights. The causes of this inattention can be

diverse, depending on factors such as perception, attention and emotion problems (Dehais *et al.*, 2019). Due to this decreased attention in the cockpit, it becomes crucial to provide cognitive assistance to pilots (Zhang *et al.*, 2018). This paper presents a proposal for a cognitive synthetic pilot acting as a coach. It is based on the ACT-R (Adaptive Control of Thought—Rational) cognitive architecture and integrates an ontology defining standard piloting procedures as a knowledge base. The aim is to provide recommendations to novice pilots during simulator training.

The ontological reference model used is described in (Courtemanche *et al.*, 2022). It consists of production rules closely linked to the execution context, providing a framework for automatic problem solving. This means that normal and abnormal procedures are decomposed into production rules to identify the actions expected of the pilot in each situation. This automatic resolution is achieved by integrating environmental parameters into the reference framework. The reference model is structured into domain and task ontologies. The domain ontology groups the terminology related to the execution environment, mainly aiming to facilitate execution in a complex context. The task ontology provides a taxonomy of aeronautical procedures specific to the aircraft piloting domain.

Based on Newell's criteria (Newell, 1990) and Sun's desiderata (Sun, 2004), we have opted for the ACT-R cognitive architecture. The principles described by these criteria and desiderata provide guidelines for evaluating and designing relevant, credible and robust cognitive models. ACT-R is used to develop cognitive models that simulate the functioning of the human brain in various tasks, from problem-solving to learning and decision-making (Anderson *et al.*, 2004). These models help better understand the underlying mechanisms of cognitive processes and predict human behavior in specific contexts such as aircraft piloting (Kotseruba et Tsotsos, 2020). ACT-R is a production system composed of a declarative memory as well as a procedural memory. The former contains knowledge, while the latter contains production rules. The declarative module deals with recognizing what is presented to the model and calculating rule activation, while the procedural part deals with calculating the utility of each activated rule and triggering the most appropriate one. Cognition emerges from the interaction between procedural and declarative structures (Anderson, 2007).

Finally, the paper outlines the methodology used to build the synthetic pilot, highlighting the integration of the ontological reference model and the production rules. Thus, in order to ensure the proper functioning of our synthetic pilot, we decided to store the domain ontology and task ontology in its declarative memory.

As for the rules used to make inferences about knowledge, such as production rules and SWRL (Semantic Web Rule Language) rules, we have stored them in the procedural memory of our ACT-R cognitive agent. This structuring allows him to have the information he needs to reason and make decisions autonomously in his environment. The decisions made by the cognitive agent are modeled by a 3D directed graph. This graph is dynamically generated over time and displayed in the visual terminal, allowing the synthetic pilot to identify the next task to be performed and provide a visual explanation of the proposed recommendation in the form of a path through the directed graph. To build this directed graph dynamically, the cognitive agent must always know the final goal to be achieved and the current task. The built cognitive pilot is able to detect deviations from standard procedures and provide contextual assistance to the novice pilots in the form of alerts, comments and recommendations.

The results obtained show that cognitive coaching based on the ontological reference model can improve the learning of procedures and accelerate the acquisition of automatisms in novice pilots, compared with unassisted training.

This research opens up new perspectives in designing expert training systems in virtual environments where, in addition to the expert knowledge provided by the ontology, machine learning can be used to increase the cognitive agent's knowledge by learning pilots' behavioral profiles from real flight data.

2 Related Work

Several works, such as the ontological representation of knowledge in the aircraft field, the automatic execution of ontological piloting procedures, the automatic flight deviation detection, as well as the execution of a piloting task by an ACT-R agent, have laid the foundations for the development of our cognitive synthetic pilot.

2.1 The Reference Model

In order to formally represent the knowledge related to the task, the task environment, and pilot cognitive processes, ontologies can be used. Mizoguchi and Bourdeau (2000) have shown the benefits of using an ontological structure to represent rules in order to formally model the knowledge of a domain. In the aeronautical field, ontologies have been developed to formalize knowledge representation. Among these, notable research has been conducted by Yousefzadeh Aghdam *et al.* (2021), Stefanidis *et al.* (2020), as well

as Sheng *et al.* (2020). Although these works have contributed to formalizing knowledge, none of them has actively exploited expert domain knowledge, formalized through an ontology, with the aim of directly assisting pilots and providing them with immediate feedback on task execution. The ontological reference model proposed in (Courtemanche *et al.*, 2022), addresses the challenge of integrating pilots' expertise into an ontology to actively support the execution of flight tasks.

The ontological reference model formalizes expert knowledge of piloting procedures using two linked ontologies developed in OWL (Web Ontology Language) : a task ontology that captures in a structured way the sequential, temporal and contextual representation of procedures acquired from expert pilots, and a domain ontology that describes the cockpit environment. This two-step process based on standard formalisms allows for the formal and detailed modeling of these complex procedural knowledge for automated exploitation by a cognitive agent (Figure 4.1).

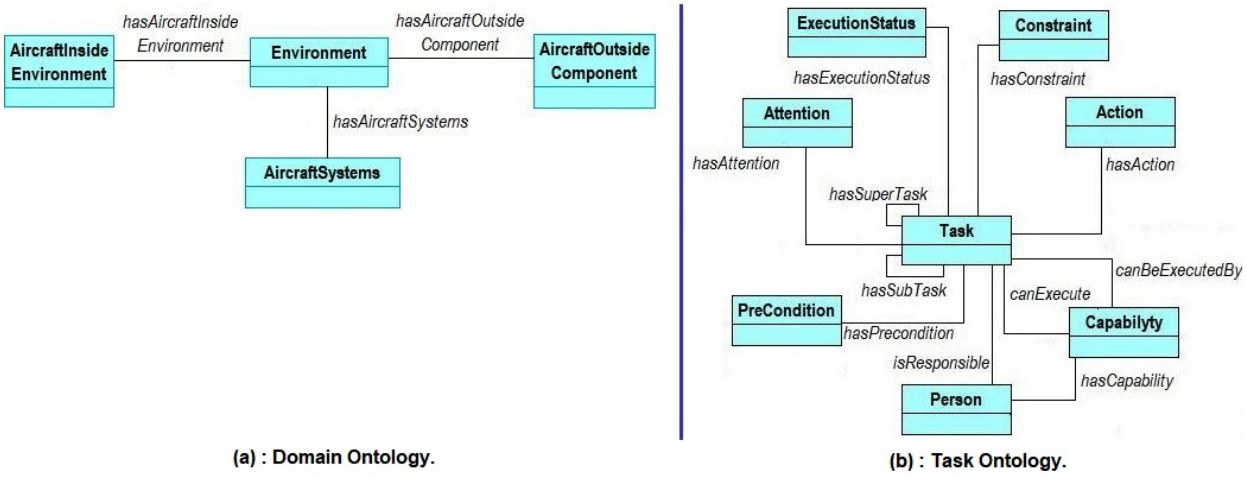


Figure 4.1 – Ontological reference Model.

2.2 Automatic Execution of Reference Model

Except for the approach proposed in (Courtemanche *et al.*, 2023), there is no documented solution in the literature for automating the execution of the ontological reference model. The authors of this paper have proposed a framework for the automatic interpretation and execution of the production rules contained in the reference model. This execution framework uses the procedural and declarative knowledge available in the model. The interpreter uses semantic rules to evaluate environmental constraints, task preconditions and task execution, enabling automatic execution. The simulator used includes a reasoner that manipulates knowledge, evaluates the task and aircraft environment, and autonomously executes required tasks. To

support the execution model, the reasoner uses the Semantic Web Rule Language. The set of rules thereby makes the reference model automatically executable.

2.3 Automatic Flight Deviation Detection

Pietracupa and colleagues focused on the automatic detection of flight deviations using the ontological reference model for piloting procedures (Pietracupa *et al.*, 2023).

This paper presents an innovative system, aimed at detecting in real time deviations in the actions performed in the cockpit from the reference procedures established in the reference ontology for pilot procedures. The aim is to anticipate and prevent potential errors that may occur when processing complex data in very short periods, typical of aircraft piloting. To assess these deviations, the authors have developed a model that uses the Needleman-Wunsch global alignment algorithm to compare pilots' actions with reference sequences defined in the ontology. They also integrated a Siamese LSTM network, a type of recurrent neural network, to understand the relationships between different action sequences. This approach facilitated the detection of errors such as added, omitted, or incorrectly ordered actions. This system is limited to detecting deviations from aviation experts' prescriptions, without cognitive modeling or recommendation formulation.

2.4 Execution of a Piloting Task by an ACT-R Agent

Other than the research published in (Tamkodjou Tchio *et al.*, 2023), we have not found any documentation in the literature dealing with the integration of the ontological reference model of piloting procedures with an ACT-R cognitive agent.

The paper presents a cognitive agent based on the ACT-R cognitive architecture, which integrates an ontological reference model of the aircraft domain to simulate complex piloting tasks. This reference model is part of the ACT-R agent's declarative memory, while the production rules used to extract knowledge are stored in its procedural memory.

The study also introduces an algorithm detailing how the cognitive agent executes a task. Finally, the paper describes the methodology used for implementation, using the pyactr library for the ACT-R model and owlready2 for ontology-oriented programming.

Experimental results demonstrate the successful execution of a piloting task by the cognitive agent, similar to the cognitive cycle observed in human pilots.

3 The Cognitive Synthetic Pilot

The cognitive synthetic pilot presented here is based on the ACT-R cognitive architecture and integrates the ontological reference model into its declarative memory. The choice of ACT-R was guided by Newell's criteria and Sun's desiderata, which establish various capabilities, properties and evaluation criteria for cognitive architectures (Kotseruba et Tsotsos, 2020). Cognitive architectures are a significant research topic in cognitive psychology, philosophy of mind, artificial intelligence, and cognitive sciences. The ACT family of architectures was developed by John Robert Anderson in 1973 with the aim of providing a comprehensive theory of human cognition (Anderson *et al.*, 2004). Kotseruba and Tsotsos (2020) define a cognitive architecture as the composition of computer tools aimed at generating perception, reflection, and decision-making capacities similar to those of a human being. The ultimate goal of cognitive architectures is to revive the dream of strong artificial intelligence (Russell et Norvig, 2020; Lieto *et al.*, 2018). ACT-R (Adaptive Control of Thought - Rational) (Anderson *et al.*, 2004) is a cognitive architecture that aims to simulate and understand human cognition through a set of modules representing different cognitive processes. ACT-R's main modules are :

- Perceptual modules, receiving sensory information ;
- Motor module, controlling motor actions ;
- Declarative module, storing factual knowledge in long-term memory ;
- Procedural module, containing production rules for action selection ;
- Coordination module, selecting production rules and modules to be activated.

Cognitive functioning in ACT-R is based on chunks of information that circulate between modules according to a cognitive cycle : perception, memory retrieval, motor action. Our synthetic pilot, based on the ACT-R model, operate as follows :

- The declarative module contains knowledge provided by the reference model. It includes the domain ontology and task ontology (Retrieval Buffer). It also stores information about the current task state (Imaginal Buffer) and the goal to be achieved (Goal Buffer) ;
- The procedural memory models productions (condition-action-states) based on the current state of the piloting task or the aircraft environment. It also contains SWRL rules to automate task execution and determine their execution status ;

— Perceptual modules (Visual Buffer) and motor modules (Manual Buffer) model basic sensory processes and interact with the environment.

Based on information captured in the environment, the coordination module or pattern matcher queries the reference model to deduce whether there is a deviation or not. In the case of a deviation, it proposes actions to manage it.

To successfully complete its mission, the synthetic pilot remains flexible and adaptive during task execution. It also knows the operational context of the ongoing task and related tasks. It also understands what constitutes normative performance for a task and the means to achieve the desired goal. The operating architecture of our synthetic pilot is shown in Figure 4.2.

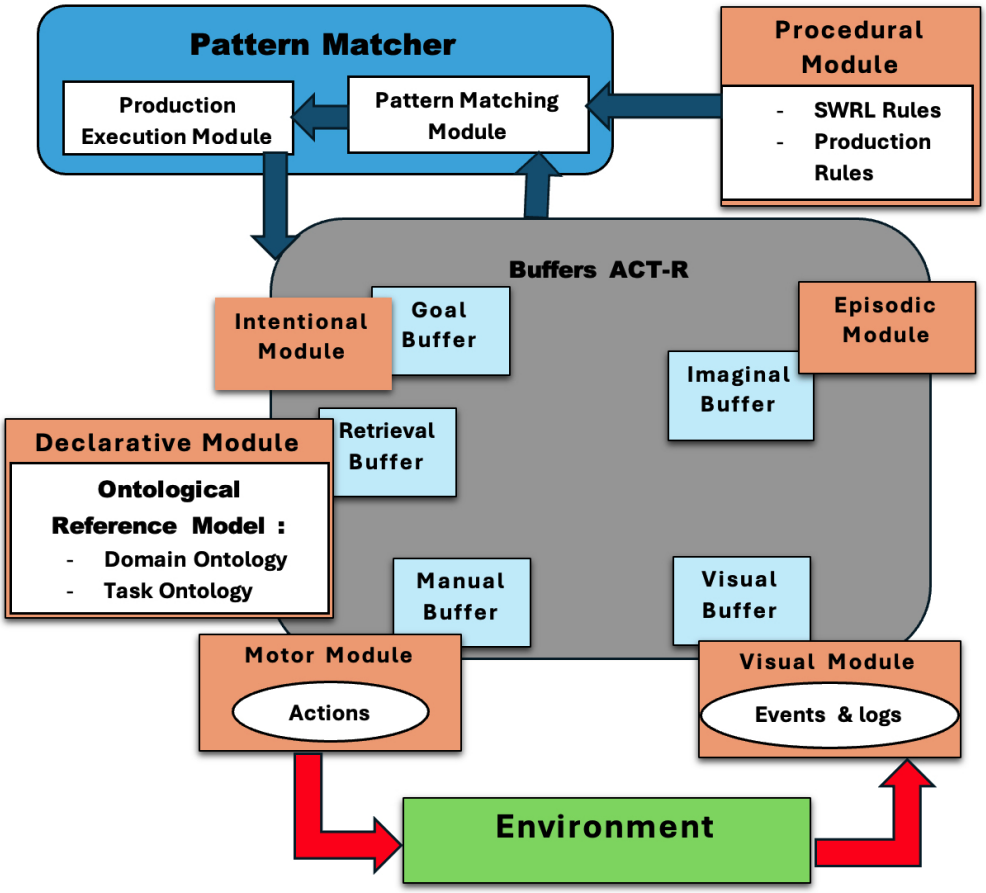


Figure 4.2 – Synthetic pilot internal architecture.

4 Methodology and Results

4.1 Methodology

To build the synthetic cognitive pilot, we relied on the following tools :

- The Pyactr Python library, which allowed us to create our cognitive agent based on the ACT-R cognitive architecture. Pyactr provides tools to define cognitive models, specify production rules, and simulate cognitive processes such as perception, memory and action ;
- The owlready2 Python library, which enabled us to manipulate the ontological reference model (domain and task ontologies). Owlready2 allows representing an ontology, accessing and modifying its classes and properties ;
- The Pellet reasoning engine, which facilitated logical deductions on the domain and task ontologies ;
- The Semantic Web Rule Language (SWRL), which enabled us to write logical rules to infer task execution status from information extracted from the ontological reference model ;
- The NetworkX library enabled representing the network of tasks as a graph with nodes and edges. We could then use NetworkX functions to generate and update the cognitive agent's path through this task network graph over time. The library provided the necessary capabilities to construct, navigate, and analyze the agent's trajectory in the complex, evolving task network. Overall, NetworkX facilitated modeling and tracking the cognitive agent's dynamic pathway within the complex network of tasks. The Pyvis library enabled interactive and dynamic network visualization of the complex graphs created with NetworkX. We could leverage Pyvis to generate web-based visualizations of the node-edge graphs and interact with the network by zooming, dragging, and hovering over elements. This interactivity facilitated analysis and interpretation of the complex networks. Subsequently, Matplotlib empowered 3D visualization of the network graphs, enhancing our ability to understand and present the high-dimensional graph structures. The 3D plots provided more visually intuitive representations of the networks compared to 2D plots. Overall, Pyvis and Matplotlib complemented NetworkX to enable both interactive exploration and 3D visualization of the complex task networks ;
- The Protégé Ontology Editor, with its user-friendly graphical interface for manipulating reference model elements (classes, properties, individuals, etc.) ;
- The Unified Modeling Language (UML), used to visually represent the concepts and relationships in the reference model.

The sequence diagram in Figure 4.3 models, in a temporal order, the interactions within the cognitive agent during the execution of a task.

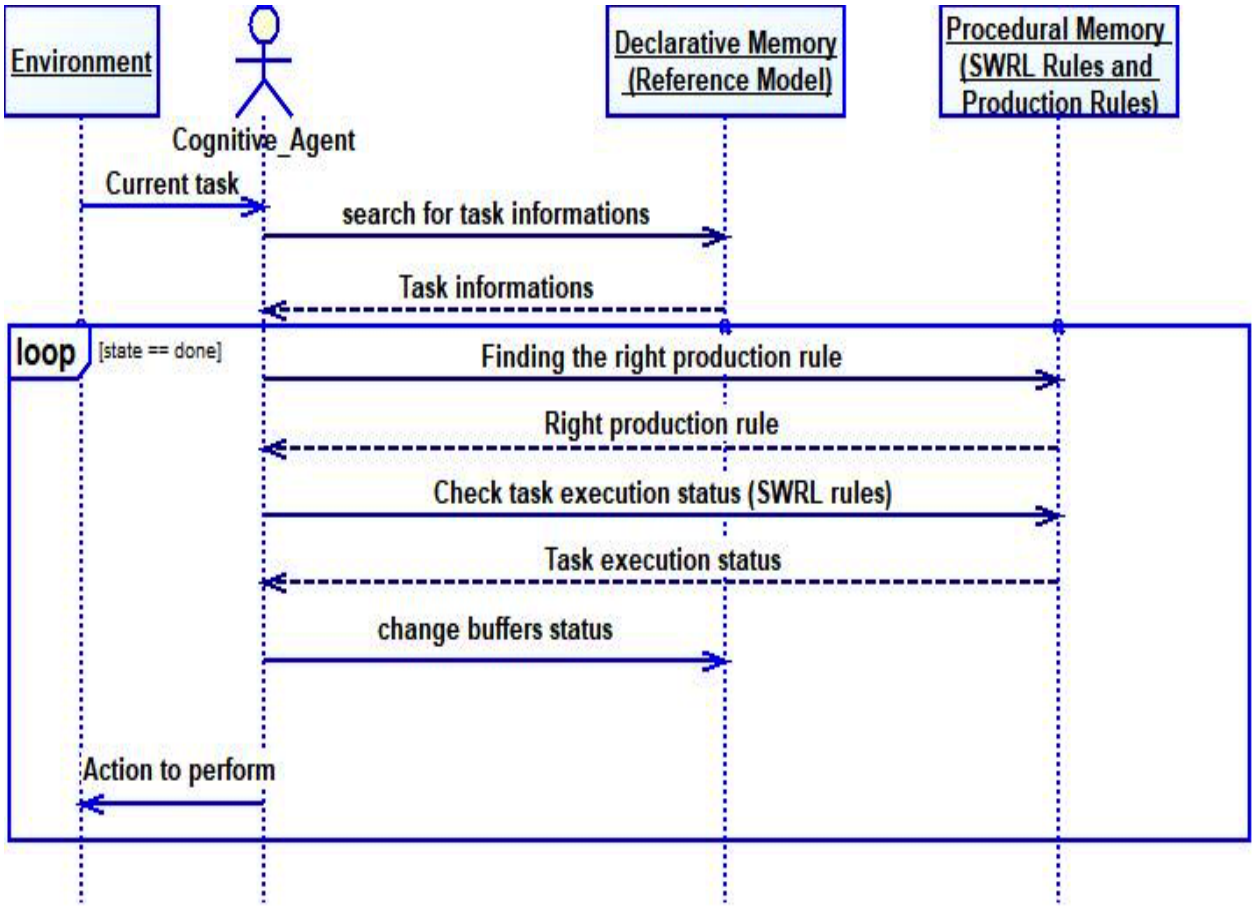


Figure 4.3 – Sequence diagram of the execution of a task by the synthetic cognitive pilot.

Algorithm 1 shows how the synthetic pilot performs the takeoff procedure described in the ontological reference model.

To make the reference model directly executable, we have defined several SWRL rules. Two examples of which are presented in Figure 4.4. The first rule (Figure 4.4.a) is designed to validate tasks with an action and a precondition (for example : task 1001). The second rule (Figure 4.4.b) is intended to validate tasks with a constraint of type 1 and another of type 5 (for example : task 1019).

Algorithm 1 : Algorithm of takeoff procedure

Input : Ontological Reference Model O_M

Parameter : Takeoff T_O, Current Task C_T, Objective O_B

Output : Execution Time E_T, Directed Graph D_G, Task T

1: O_B = goal(T_O (Set takeoff as the goal.)
2: C_T = 1000 (The current task is set as the initial task (1000).)
3: status = Executed (The current task status is set to Executed.)
Repeat the actions below until the current task is task 1035 (last task of Takeoff)
4: **repeat**
5: D_G = generated_path(D_G,O_B) (Generate the directed graph with the current task (C_T) as the starting node and task 1035 as the ending node.)
6: print(D_G) (Display the directed graph in the visual interface.)
7: T = next_task(D_G) (Select the new current task in the directed graph D_G.)
8: status = swrl(C_T) (The status will be set to Executed if C_T was executed successfully, NotExecuted otherwise.)
9: **if** status == Executed **then**
10: C_T = T (The current task C_T is set to the next task T.)
11: E_T = pattern_matcher(C_T,O_M) (The cognitive agent uses its complex pattern matching mechanism to execute task C_T and make recommendations.)
12: print(E_T) (Display execution time E_T.)
13: **end if**
14: **until** T == T_O
15: buffers = [T_O, O_B, E_T, C_T, D_G, T]
16: **return** Buffers

```
Task(?t), canBeExecutedBy(?t,?capability), canExecute(?capability,?t),
Person(?p), isResponsible(?p, ?spt), hasCapability(?p, ?capability),
hasSuperTask(?t, ?spt1), hasExecutionStatus(?spt1, LiveExecution),
    hasAction(?t, ?act1), hasActionParameter(?act1, VerballyAnnounce),
    hasActionValue(?act1, ?action_val1), verballyAnnounce(VerballyAnnounce, ?action_val1),
hasPreCondition(?t, ?pc1),
hasExecutionStatus(?pc1, Executed)
-> hasExecutionStatus(?t, Executed)
```

(a)

```
Task(?t), canBeExecutedBy(?t,?capability), canExecute(?capability,?t),
Person(?p), isResponsible(?p, ?spt), hasCapability(?p, ?capability),
hasSuperTask(?t, ?spt1), hasExecutionStatus(?spt1, LiveExecution),
                hasConstraint(?t, ?const1), hasConstraintType(?const1, ?ctype1),
                equal(?ctype1, 5), hasEvaluationCriteria(?const1, ?ev1),
hasExecutionStatus(?ev1, Executed),
                hasEvaluationCriteria(?const1, ?ev2),
hasExecutionStatus(?ev2, Executed)
-> hasExecutionStatus(?t, Executed)
```

(b)

Figure 4.4 – Example of two SWRL constraint evaluation rules.

4.2 Results

To assess our model, we conducted several simulations. Initially, we performed complete takeoff procedures. Subsequently, we tested the synthetic pilot by having it perform takeoff recoveries at randomly chosen moments in the takeoff procedure. In each case, we dynamically generated the directed task graph and compared it to the execution expected by the ontological reference model.

During the complete procedure, the graph (Figure 4.6) generates tasks identical to those planned by the experts (Figure 4.5). Based on this graph, the cognitive agent executes the tasks appropriately.

Regarding takeoff recovery procedure, two cases can occur :

— In the first situation, the generated directed task graph (see from task 1024 in the Figure 4.7.d) is easy to navigate, and the results provided by the cognitive agent conform to the reference (Figure 4.7.a, Figure 4.7.b, Figure 4.7.c) ;

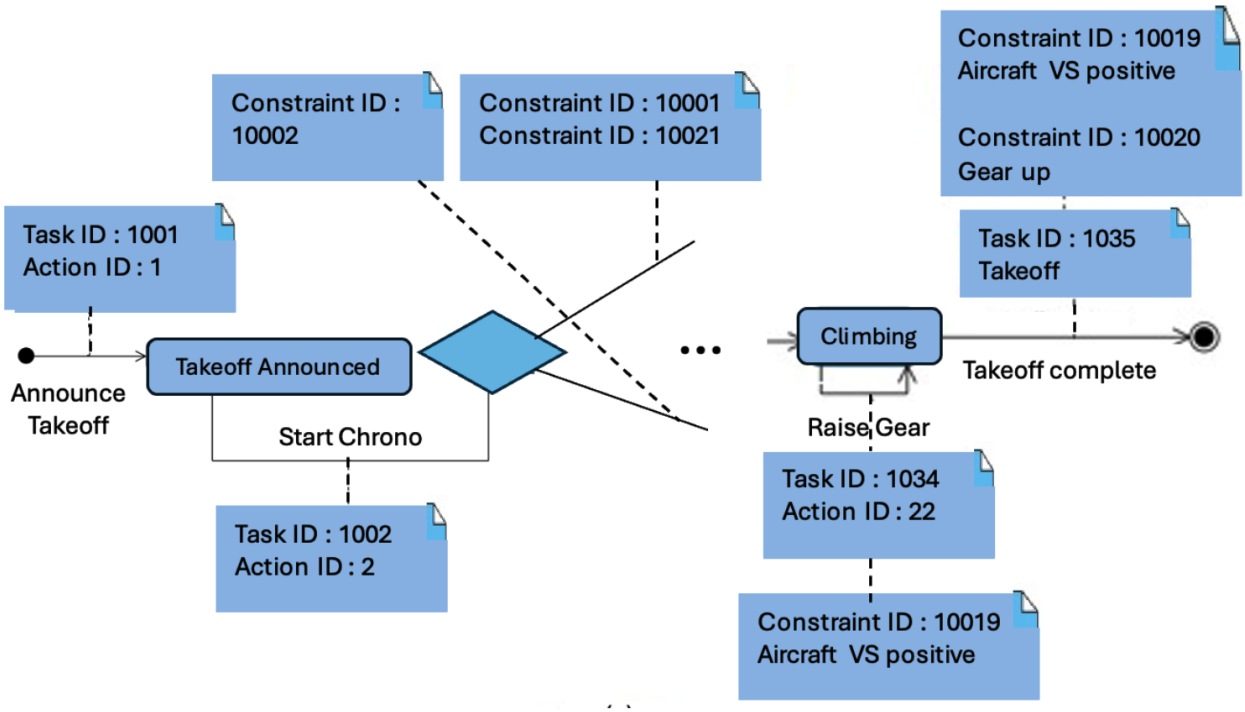


Figure 4.5 – Reference state-transition diagram.

— In the second situation, the cognitive agent may find itself in a deadlock (tasks 1025 and 1026), meaning a situation where the ontology has not provided any solution (Figure 4.7.a, Figure 4.7.b, and Figure 4.7.c). To exit this deadlock, we search the ontology for a neighboring task leading to a solution and generate a link to that task (Figure 4.7.d). If this is not possible, we end up with a crash. Adding new links allows the agent to complete its task.

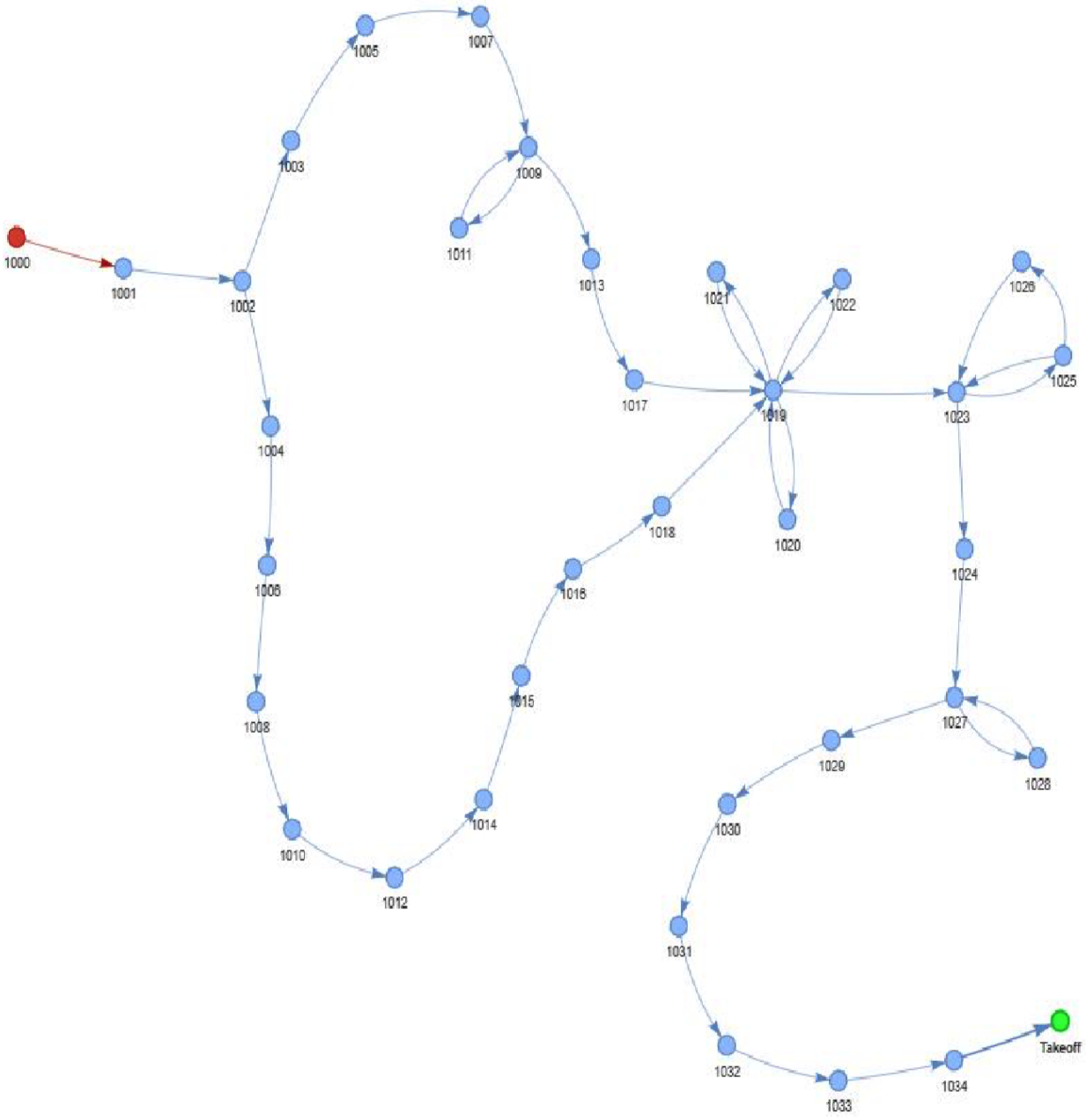


Figure 4.6 – Directed task graph for the takeoff procedure.

The agent can function in two modes (Figure 4.8) : an autonomous mode and an interactive mode. In autonomous mode, it performs the procedure alone, presenting its results on the terminal (Figure 4.8.b). In interactive mode, it operates interactively, makes recommendations to avoid deviations (Figure 4.8.a), acting as a coach assisting the pilot in his task.

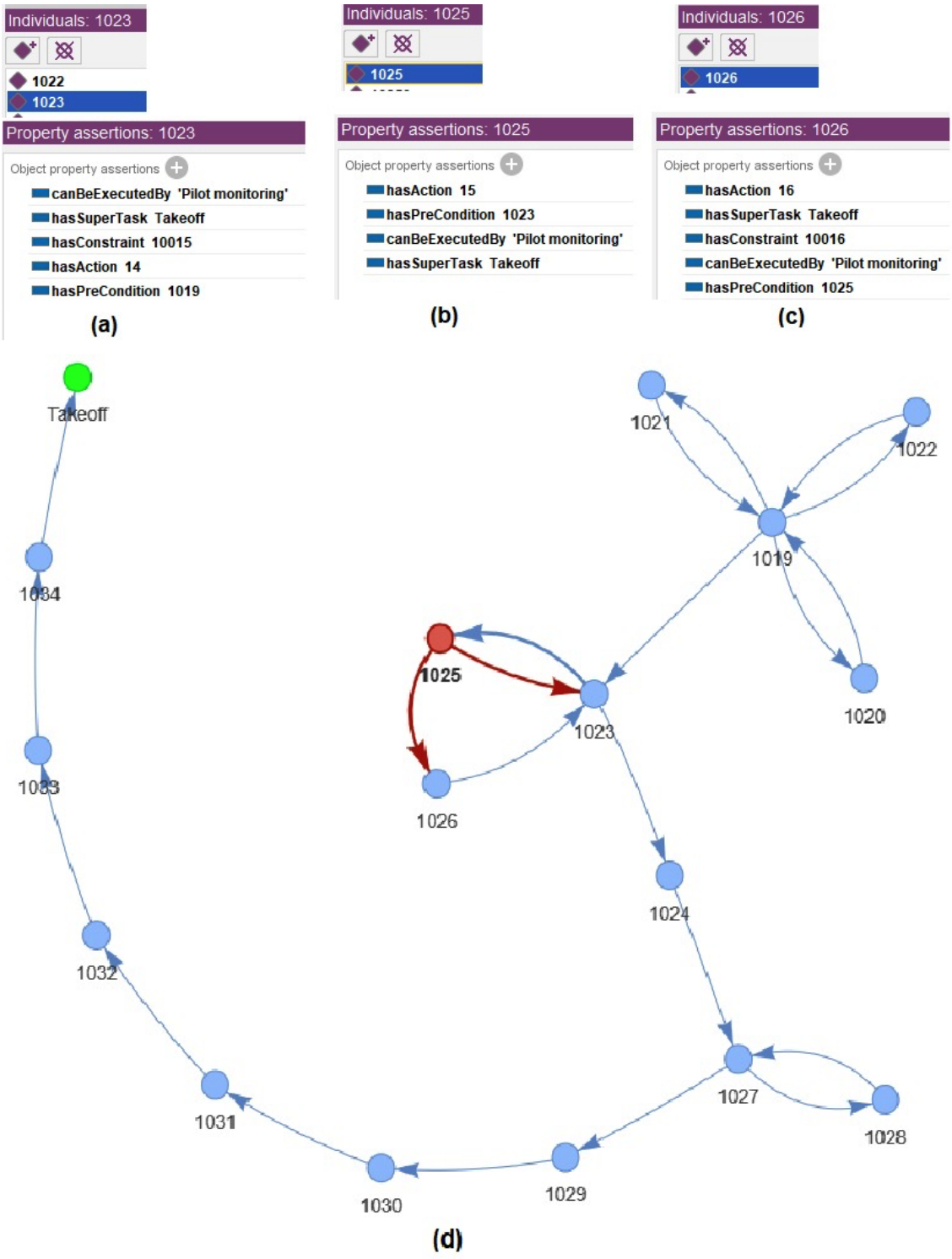
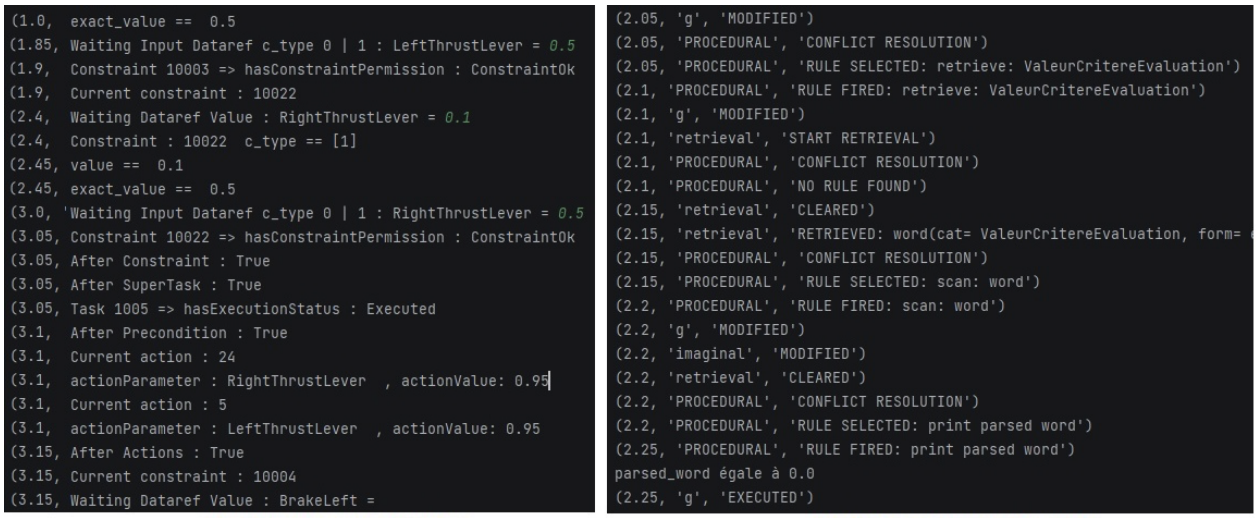


Figure 4.7 – Takeoff recovery : tasks involved from the reference model and directed task graph with added new links.



(a) : Interactive execution mode **(b) : Autonomous execution mode**

Figure 4.8 – Execution modes of the synthetic pilot.

5 Conclusion and Future Works

In this paper, we proposed a synthetic pilot based on the ACT-R cognitive architecture and using an ontological reference model as a knowledge base. This reference model includes a domain ontology that captures and formalizes the internal, external environments and navigation systems of aircraft, and a task ontology that captures and formalizes piloting procedures. To carry out its mission, the synthetic pilot uses an interpreter made up of SWRL rules, allowing automatic manipulation of the knowledge provided by the ontological reference model. In its quest for a solution, it dynamically generates and displays a 3D directed graph, which it navigates to determine the tasks to be executed.

Experiments have demonstrated the agent's ability to perform a complete takeoff procedure, similar to a human expert pilot. Furthermore, the agent can execute a takeoff recovery and reconfigure, in certain cases, a path in the form of a directed graph not foreseen in the reference model to achieve the set objective. Finally, the cognitive agent can be used in two modes : an autonomous mode in which it carries out the procedure from A to Z, presenting the different steps, and an interactive mode where it provides recommendations to the human pilot. In this way, the pilot can perform the actions recommended by the system to avoid deviations.

The next stage of this study will be to validate our proposal by testing the synthetic pilot in real-life situations, for example on the X-Plane flight simulator. Another focus will be to develop a synthetic pilot able to

perform procedures related to abnormal tasks that occur during takeoff, such as "Dual engine failure with fuel remaining", "Engine failure after V1", "Reactive Windshear", "Rejected Takeoff", "TCAS event" and "Stall recovery". Finally, a future perspective will focus on enhancing the understanding of our synthetic pilot by integrating knowledge from learning pilot behavioral profiles from real flight data (Tato *et al.*, 2023).

Acknowledgments. We acknowledge the support of Bombardier and all the members of the Pilot-AI project.

4.3 Discussion et Conclusion

Dans ce chapitre, nous avons étendu les capacités du pilote synthétique afin qu'il puisse prendre en charge la procédure standard de décollage en situation normale. L'approche proposée allie la modélisation des processus cognitifs humains à une représentation formelle des connaissances aéronautiques, notamment les procédures de pilotage d'un avion, afin de concevoir un système de coaching capable d'assister les pilotes dans l'exécution de leurs tâches. Le pilote synthétique ainsi développé se distingue par sa capacité à reproduire les mécanismes cognitifs humains via l'architecture ACT-R, tout en s'appuyant sur la richesse sémantique et procédurale de l'ontologie de référence, structurée en deux volets complémentaires : une ontologie du domaine capturant l'environnement du cockpit, et une ontologie des tâches formalisant les procédures de pilotage.

Cette combinaison permet à l'agent de structurer efficacement ses connaissances et d'automatiser son raisonnement, en s'appuyant sur un moteur d'inférence et un ensemble de règles dérivées de la sémantique ontologique. Elle lui permet également de raisonner sur les connaissances aéronautiques et d'exécuter les tâches selon un cycle cognitif similaire à celui d'un pilote humain expert.

L'implémentation technique de la solution s'est appuyée entre autres sur plusieurs outils clés : la bibliothèque Pyactr pour la modélisation cognitive, Owlready2 pour la manipulation des ontologies, le moteur de raisonnement Pellet pour les déductions logiques, et le langage SWRL pour l'écriture de règles sémantiques. L'utilisation combinée de ces technologies a permis de développer un agent capable non seulement d'exécuter des procédures de décollage standards de manière conforme aux pratiques expertes, mais aussi de détecter en temps réel les déviations et de proposer des actions correctives pertinentes adaptées au contexte.

Les résultats des simulations ont démontré l'efficacité du pilote synthétique à fonctionner selon deux modes

distincts : un mode autonome, où l'agent exécute seul les procédures standard, et un mode interactif, où il joue le rôle de coach ou d'assistant, guidant le pilote humain à travers des recommandations contextuelles. Dans les deux cas, l'agent génère dynamiquement un graphe orienté des tâches, qu'il utilise pour naviguer à travers la procédure, identifier les actions à entreprendre et offrir une visualisation claire des étapes à suivre.

Un apport significatif de notre approche réside dans la capacité de l'agent à gérer des situations imprévues, notamment lorsqu'il se retrouve dans une impasse non prévue par le modèle de référence. En analysant l'ontologie, il est capable d'identifier une tâche voisine menant à une solution et de générer un nouveau lien vers celle-ci, démontrant ainsi sa flexibilité et son adaptabilité face à des scénarios complexes ou inattendus.

Les résultats obtenus indiquent que, pour la phase de décollage en situation normale, le coaching cognitif fourni par le pilote synthétique peut faciliter l'apprentissage des procédures standard et accélérer l'acquisition des automatismes chez les pilotes novices, en comparaison avec un entraînement sans assistance (Zhao *et al.*, 2024; Kozak, 2019).

Toutefois, certaines limites subsistent et ouvrent des perspectives de recherche futures. Le système pourrait notamment être étendu pour gérer des situations anormales survenant lors du décollage, bien qu'il soit actuellement centré sur la procédure standard. Une telle expérimentation permettrait de traiter des événements critiques tels qu'une panne moteur après V1, un cisaillement de vent réactif, une alerte de trafic ou d'évitement de collision, entre autres. Cet aspect sera exploré en détail dans le chapitre suivant.

CHAPITRE 5
VERS UNE ASSISTANCE COGNITIVE AVANCÉE POUR LE PILOTAGE : EXTENSION DU PILOTE SYNTHÉTIQUE ACT-R À LA GESTION DES PROCÉDURES ANORMALES AU DÉCOLLAGE

5.1 Préambule

Le chapitre précédent a présenté comment le pilote synthétique basé sur ACT-R pouvait exécuter une tâche de pilotage d'un avion, en l'occurrence un décollage en conditions normales, en suivant les prescriptions d'experts de l'aviation contenues dans l'ontologie de référence.

Dans notre article « *Handling of Abnormal Aircraft Takeoff Procedures : Cognitive Modeling of an ACT-R Synthetic Pilot Integrating an Ontological Reference Model* » (Tamkodjou Tchio *et al.*, 2024b), nous avons démontré comment le pilote synthétique peut gérer efficacement les procédures anormales lors de la phase critique du décollage. Pour davantage de détails sur cette gestion des procédures anormales, nous invitons le lecteur à consulter l'article de Tamkodjou Tchio *et al.* (2024b).

En effet, bien que le décollage ne représente qu'environ 2% de la durée totale d'un vol, il concentre près de 30% des accidents mortels (Clifford, 2022). Parmi les situations critiques pouvant survenir à cette phase figurent les pannes moteur après V1, les décollages interrompus, les cisaillements de vent réactifs, les événements TCAS, les doubles pannes moteur avec carburant restant, et les récupérations de décrochage.

Dans ce chapitre, nous élargissons les capacités du pilote synthétique ACT-R afin qu'il puisse gérer les décollages, qu'ils relèvent de procédures normales ou anormales, conformément aux standards prescrits par les experts de l'aviation. Cette approche vise à améliorer la formation des pilotes et à renforcer leur prise de décision, deux éléments essentiels à la sécurité aérienne (Tamkodjou Tchio *et al.*, 2025).

L'ontologie de référence utilisée par le pilote synthétique pour la prise de décisions résulte de la fusion de l'ontologie de gestion du trafic aérien (ATM - Air Traffic Management) développée par la NASA (Keller, 2017b) et du modèle de référence ontologique proposé par Courtemanche *et al.* (2022, 2023), avec des adaptations spécifiques à notre contexte. Elle se compose toujours d'une ontologie du domaine et d'une ontologie des tâches.

Ce chapitre est structuré en plusieurs sections qui détaillent le développement du pilote synthétique cognitif.

Le chapitre commence par présenter les fondements théoriques sur lesquels repose l'approche, mettant en avant les principes de l'architecture ACT-R et de l'ontologie de référence qui constituent la base de notre approche d'assistance cognitive.

Une revue des travaux connexes est ensuite proposée. Elle porte sur la représentation ontologique des connaissances dans le domaine aéronautique, l'intégration des ontologies dans les architectures cognitives, ainsi que l'usage de l'assistance cognitive dans les contextes de formation et de prise de décision.

L'ontologie de référence utilisée par le pilote synthétique est ensuite présentée. Ce modèle formalise la connaissance experte sur laquelle se base l'agent cognitif pour gérer des procédures complètes de décollage en situations normale et anormale. La Figure 5.1, illustre le diagramme de classes de l'ontologie ATM.

La suite du chapitre décrit l'architecture du pilote synthétique cognitif, en détaillant ses principaux composants : mémoire déclarative, mémoire procédurale, modules perceptifs et moteurs, ainsi que l'intégration des règles SWRL pour l'exécution automatique des tâches. La Figure 5.2 présente l'architecture opérationnelle du pilote synthétique, offrant une vue d'ensemble de l'interaction entre ces modules pour simuler les processus cognitifs du pilotage et fournir une assistance personnalisée.

La méthodologie d'implémentation et les résultats expérimentaux sont aussi abordés. L'implémentation technique du pilote synthétique est décrite à l'aide des bibliothèques Python PyACTR et Owlready2. L'algorithme utilisé (Algorithme 1), les outils technologiques mobilisés, et les résultats des simulations réalisées dans divers scénarios de décollage, qu'ils soient normaux ou critiques, y sont également détaillés.

Les deux modes de fonctionnement du pilote synthétique, autonome et interactif, sont également mis en évidence, offrant ainsi une flexibilité précieuse pour la formation et l'aide à la décision, comme l'illustrent les Figures 5.8 et 5.9.

Enfin, le chapitre se termine par une synthèse des contributions et une discussion sur les perspectives futures, incluant notamment l'extension des capacités du pilote synthétique par l'usage de l'apprentissage machine pour intégrer des données réelles issues des vols et la prise en considération des enjeux éthiques

liés à l'utilisation de la plateforme de formation.

Comme dans le chapitre précédent, nous avons utilisé la Distance d'Édition de Graphe (Graph Edit Distance — GED) comme métrique pour évaluer la validité des résultats produits par le pilote synthétique. La GED est une mesure fondamentale en théorie des graphes, car elle permet de quantifier la similarité structurelle entre deux graphes en calculant le coût minimal nécessaire pour transformer l'un en l'autre à l'aide d'un ensemble d'opérations élémentaires telles que l'insertion, la suppression ou la substitution de nœuds et d'arêtes. Cette métrique est particulièrement adaptée à notre contexte, où l'on cherche à comparer le graphe des tâches générés par le pilote synthétique au graphe de référence établi par les experts du domaine.

5.2 Enhancing Pilot Training and Decision-Making Using Ontologies : A Cognitive Assistance Approach

La suite de ce chapitre présente l'article intitulé « *Enhancing Pilot Training and Decision-Making Using Ontologies : A Cognitive Assistance Approach* », écrit par *Guy Carlos Tamkodjou Tchio*, *Roger Nkambou*, *Valéry Psyché* et *Ange Adrienne Nyamen Tato*. Cet article a été publié dans l'ouvrage « *Generative Systems and Intelligent Tutoring Systems (ITS 2025)* » et présenté lors de la *21st International Conference on Intelligent Tutoring Systems (ITS'2025)*, qui s'est tenue à Alexandroupolis, en Grèce, du 02 au 06 juin 2025.

Abstract. This paper presents a cognitive assistance approach using ontologies to enhance pilot training and decision-making. A reference ontology, combining a domain ontology representing the aircraft's external environment and a task ontology describing flight procedures, is leveraged. Based on this, a synthetic pilot, developed using the ACT-R cognitive architecture, provides personalized cognitive assistance to pilots, helping them in their training and decision-making processes. The synthetic pilot adapts to individual learning preferences and progress, offering tailored feedback and guidance. The aim is to increase pilots' skills and enhance aviation safety through this personalized ontological cognitive assistance.

Keywords : Ontologies · Domain Ontology · Task Ontology · ACT-R · Synthetic Pilot · Cognitive Assistance · Modeling · SWRL Rules · Graph Theory

1 Introduction

Pilot training and decision-making are crucial for flight safety, despite technological advances and persistent challenges (Dismukes *et al.*, 2007). Ontologies, by enabling a formal and shared knowledge representation (Gruber, 1993), help enhance training and decision-making in critical situations (Insaurralde et Blasch, 2022).

This paper proposes an innovative ontology-based cognitive assistance approach to improve pilot training and decision-making. This approach is based on the use of a reference ontology combining two aspects : a domain ontology describing the external environment of the aircraft as well as the execution environment of complex piloting tasks, and a task ontology modeling piloting procedures as well as expert knowledge in the aviation domain. On this ontological basis, a synthetic pilot based on the ACT-R cognitive architecture, recognized for its ability to faithfully model human behavior in complex environments (Salvucci, 2006), is developed using the PyACTR Python library.

In this context, our research addresses the following research questions : Can a synthetic pilot that simulates human cognition and relies on expert knowledge of aircraft piloting procedures effectively assist a human pilot in performing their tasks, thereby enhancing their training and decision-making capabilities ? How can such a synthetic pilot be built ?

To answer these questions, we propose three hypotheses. First, a reference ontology can support the modeling of complex procedural knowledge related to aircraft piloting tasks, detailing the necessary information associated with the execution environment for each task. Second, the ACT-R cognitive architecture can effectively model human pilot behavior. Third, integrating a reference ontology of piloting procedures into an ACT-R-based cognitive agent can effectively assist human pilots, while enhancing their training and decision-making capabilities.

The synthetic pilot developed based on these hypotheses aims to provide cognitive assistance to human pilots, helping them in their learning and decision-making processes.

The reference ontology, based on NASA's Air Traffic Management ontologies (Keller, 2017a, 2018, 2017b) and those by Courtemanche *et al*. (2022), has been adapted for our context and will serve as the synthetic pilot's declarative knowledge base, while procedural knowledge will be managed through production rules

and Semantic Web Rule Language (SWRL) rules for automatic task execution.

2 Related Work

While it's impossible to cover every aspect, we've listed below a few of the works that have laid a solid foundation for our research.

Wu *et al.* (2014) propose an ontology-based approach to model aircraft and improve maintenance management systems without addressing pilot training. Insaurralde et Blasch (2016) suggest an ontology-based knowledge representation for decision-making in avionics, but it does not cover pilot training or use cognitive architecture. Yousefzadeh Aghdam *et al.* (2021) develop an ontology for air safety messages to enhance risk analysis and controller training, but it lacks a cognitive approach to decision-making. Stefanidis *et al.* (2020) present the ICARUS ontology, which is used for general aviation but does not integrate cognitive assistance for pilot training or decision support. Insaurralde et Blasch (2018, 2019, 2022) propose an ontology-based approach for advanced air traffic management decision-making, integrating cognitive architecture, but it does not focus on pilot training or simulating human cognitive processes like ACT-R. Salvucci (2006) applies the ACT-R cognitive architecture to model driver behavior, potentially applicable to pilots, but does not use ontologies. Oliver *et al.* (2019) exploit ACT-R for pilot decision-making in emergencies but do not use ontologies. Oltramari et Lebiere (2011) integrate ontologies with ACT-R for problem-solving without addressing training or decision-making.

The works presented above have contributed to the formalization of knowledge in various domains, but none have actively leveraged the aviation expertise formalized in an ontology to provide cognitive assistance to pilots during their training and decision-making. The research conducted by Tamkodjou Tchio *et al.* (2024b, 2023, 2024a) addresses this challenge by integrating pilots' expert knowledge within an ontology to provide cognitive support to pilots during flight task execution. More precisely, Tamkodjou Tchio *et al.* (2023) introduce a cognitive agent based on the ACT-R cognitive architecture, integrating an ontological reference model to simulate, in a manner like human pilots, a task involved in aircraft takeoff. In contrast, Tamkodjou Tchio *et al.* (2024a) present an ACT-R cognitive agent that integrates an ontological reference model to simulate the complete normal takeoff procedure prescribed by domain experts. Lastly, in (Tamkodjou Tchio *et al.*, 2024b), the authors focus on managing urgent and abnormal situations during takeoff, again combining ACT-R with an ontological reference model. The approach we propose in this paper aims to combine the three approaches presented by Tamkodjou Tchio *et al.* (2024b, 2023, 2024a) while proposing

a system that supports pilots in their training and decision-making.

3 Reference Ontology for the Synthetic Pilot

The reference ontology used by our synthetic pilot combines the Air Traffic Management (ATM) ontology developed by NASA (Keller, 2017b) and the reference model proposed by Courtemanche *et al.* (2022, 2023), with some adaptations.

3.1 NASA Air Traffic Management Ontology (atmonto)

The NASA ATM Ontology (Keller, 2017b, 2018, 2017a) formally represents air traffic management knowledge, including classes, properties, and relationships related to air navigation, aircraft, airport infrastructures, airline management, and meteorological data.

In the synthetic pilot simulation, the atmonto variant (see Figure 5.1) is used, built on the foundational ontology atmontoCore, which comprises five components : NAS (airspace structures), ATM (air traffic management), data (airport-related weather and operations), equipment (aircraft systems), and general (temporal and spatial concepts).

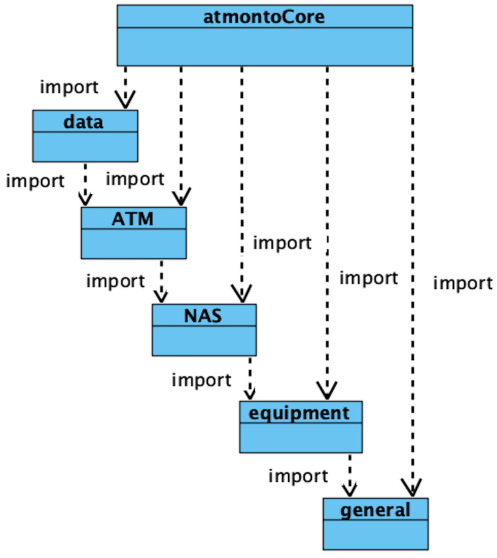


Figure 5.1 – Air Traffic Management Ontology Class Diagram.

3.2 Ontological Reference Model for Aircraft Piloting Procedures

The proposed reference model (Courtemanche *et al.*, 2022, 2023) formalizes aircraft piloting knowledge through two closely linked components : the domain ontology and the task ontology.

The domain ontology is a terminological framework specific to aircraft piloting, serving as a semantically structured data dictionary to support complex piloting tasks. It exhaustively models elements of the cockpit's internal environment, the external environment, and the aircraft's systems. Its architecture revolves around a central class, "Environment," which includes both the aircraft's interior and exterior environments. This central class includes three specialized subclasses : "AircraftOutsideComponent," representing external components ; "AircraftSystems," grouping aircraft systems ; and "AircraftInsideEnvironment," modeling the internal environment.

The task ontology, on the other hand, forms the core of the reference model. It structures the execution parameters required for complex piloting tasks. It captures key elements to model human problem-solving processes specific to the aeronautical domain. The task ontology decomposes problem-solving by integrating contextual elements closely linked to the theory formalized in the domain ontology. It provides domain-specific terminology and a framework to represent cognitive processes involved in piloting problem-solving.

Its architecture is structured around a main class , "Task," which includes several classes dedicated to specific roles. These classes define various task aspects, including the person in charge of its execution ("Person"), roles of involved personnel ("Capability"), preconditions for completion ("Precondition"), associated constraints ("Constraint"), actions on the environment and parameter values ("Action"), task workload ("Attention"), and execution status ("ExecutionStatus").

Although distinct, the domain and task ontologies are strongly interconnected through semantic links. Each constraint in the task ontology references a specific environmental parameter and its acceptable value range, as defined in the domain ontology. The same mechanism links actions in the task ontology to corresponding parameters in the domain ontology, specifying actions to be performed on the environment.

The class diagrams presenting the main groups of the domain ontology, as well as the key classes of the task ontology and their relationships, are detailed in (Tamkodjou Tchio *et al.*, 2024b, 2023, 2024a).

4 Cognitive Synthetic Pilot

Our proposal introduces a cognitive synthetic pilot, leveraging a reference ontology and the ACT-R cognitive architecture to enhance pilot training and decision-making.

The reference ontology combines NASA's ATM ontology with the ontological reference model described earlier. We have modified the rules of the ontological reference model governing access to the atmontoCore and atmonto external resources. Our reference ontology includes a domain ontology and a task ontology, enabling structured knowledge representation. The domain ontology provides vocabulary and concepts for the execution environment, while the task ontology defines problem-solving structures for tasks like piloting, supporting informed decision-making and efficient execution.

A cognitive architecture is a computational framework modeling human cognitive processes, including perception, attention, memory, reasoning, and decision-making. It offers a unified theory of cognition and can be implemented in artificial systems to replicate human-like intelligence (Kotseruba et Tsotsos, 2020; Newell, 1990; Anderson, 2007). We chose the ACT-R cognitive architecture for its proven ability to replicate human cognition. This decision aligns with the criteria outlined in Newell and Sun's desiderata (Kotseruba et Tsotsos, 2020).

ACT-R (Adaptive Control of Thought-Rational), developed by John R. Anderson and colleagues at Carnegie Mellon University in the 1970s (Anderson, 2007), is a widely studied cognitive architecture. Grounded in information processing theory, it simulates and explains human cognitive processes. ACT-R comprises interconnected modules representing cognitive functions like perception, attention, memory, reasoning, and learning. These modules collaborate to generate complex behaviors. Key modules include the visual module (sensory input), the motor module (environmental actions), the declarative module (factual knowledge), the procedural module (production rules), and the Pattern Matching module (coordination). Chunks, or information packets, enable data transfer between modules. Their flow supports the perception-reasoning-action cycle, replicating human cognition.

ACT-R has been successfully applied to model many aspects of human cognition in complex tasks such as aircraft piloting (Salvucci, 2001; Oliver *et al.*, 2019). The modular architecture of the synthetic pilot based on ACT-R is shown in Figure 5.2.

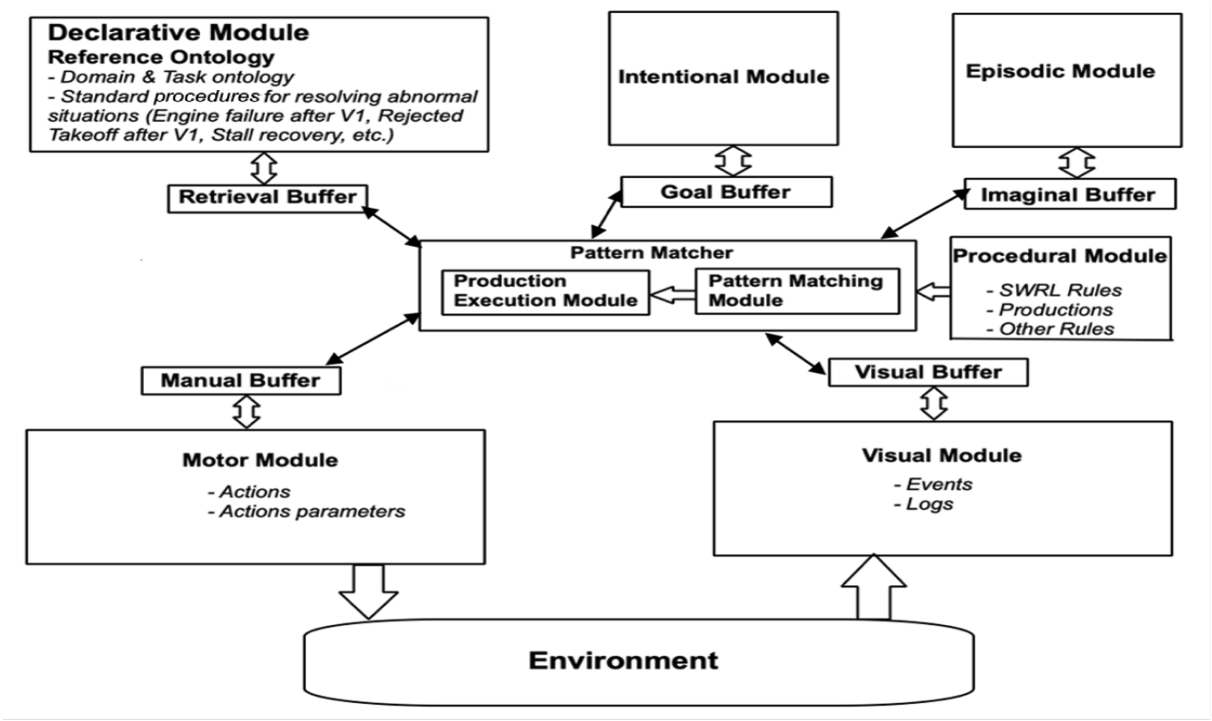


Figure 5.2 – Synthetic Pilot Operating Architecture.

Our cognitive agent comprises interconnected modules working together, each with specific functionalities :

- Declarative module : Stores and manages knowledge from the reference ontology, including domain and task ontologies, as well as the current state of the piloting task.
- Episodic module : Enables the synthetic pilot to recall past experiences, aiding decision-making, reasoning, and adaptation to current situations.
- Goal module : Directs and focuses the synthetic pilot's attention, facilitating action planning and execution to achieve task objectives.
- Procedural module : Organizes and controls actions using production rules, supporting decision-making and goal achievement as defined by the goal module.
- Perceptual module : Captures and processes sensory inputs from aircraft sensors and instruments, enabling the synthetic pilot to perceive and interpret its environment.
- Motor module : Translates the synthetic pilot's decisions into concrete actions on the aircraft.
- Pattern Matching module : Analyzes the reference ontology to detect deviations from standard takeoff procedures and error management protocols. It selects production rules and activates relevant modules to address these deviations.

With its modular ACT-R-based architecture, the synthetic pilot leverages the reference ontology to assist pilots like a human instructor, enhancing training and decision-making during normal takeoffs or incidents that occur during takeoff. The synthetic pilot manages the following incidents :

- Engine failure after V1 (Airbus, 2006a,b, 2012; Haroon, 2020; V-Prep, 2017) ;
- Rejected Takeoff after V1 (Airbus, 2006a, 2012; Haroon, 2020) ;
- Reactive Windshear at Takeoff (Airbus, 2006a, 2012; Haroon, 2020; V-Prep, 2018a) ;
- Traffic alert and Collision Avoidance System (TCAS) Event (Airbus, 2012; Haroon, 2020) ;
- Dual engine failure with fuel remaining (Airbus, 2012; Haroon, 2020; V-Prep, 2017) ;
- Stall recovery (Airbus, 2006a, 2012; Haroon, 2020; V-Prep, 2018b).

In aviation, "V1" refers to decision speed. It is a critical speed during an aircraft's takeoff phase. Specifically, V1 is the speed at which the aircraft reaches a point where the decision must be made between continuing takeoff or aborting and braking in case of emergency, such as an engine malfunction or other critical issue. Once the aircraft has reached the V1 speed, the decision to abort takeoff becomes riskier than continuing takeoff.

TCAS is an on-board collision-avoidance system used in aviation. Its main role is to monitor the airspace around an aircraft and alert pilots to other aircraft nearby that may pose a collision threat.

5 Implementation Methodology and Results

5.1 Implementation Methodology

In order to implement the cognitive synthetic pilot based on the reference ontology, we wrote the algorithm shown in Algorithm 1.

The algorithm outlines how the synthetic pilot interacts with the environment and makes decisions based on available information. The process begins by defining the main objective and determining the initial task based on the goal, such as normal takeoff (1000), engine failure after V1 (1400), rejected takeoff after V1 (1076), reactive windshear at takeoff (1251), TCAS event (1201), dual engine failure with fuel remaining (1176), or stall recovery (1350). These initial task numbers are assigned according to domain expert guidelines. A directed graph is then generated, with the current task as the starting node and the goal as the final node. The cognitive agent then successively selects the new tasks to perform in the graph. For each task, the cognitive agent uses its complex pattern matching mechanism. Relying on the knowledge formalized

Algorithm 1 : Cognitive Assistance Algorithm

Input : Reference Ontology R_O, Pilot Task to be Performed P_T
Parameter : Takeoff T_O, Engine failure after V1 EF_{V1}, Rejected Takeoff after V1 RT_{V1}, Reactive Windshear at Takeoff RW_{TO}, TCAS Event TC_{Ev}, Dual engine failure with fuel remaining DF_{FR}, Stall recovery S_R, Current Task C_T, Objective O_B
Output : Execution Time E_T, Directed Graph D_G, Task T

1: read(P_T), $O_B = goal(P_T)$ (Read and set Task to be Performed as the goal.)
2: The current task is set as the initial task (1000, 1400, 1076, 1251, 1201, 1176, or 1350).
if $P_T == T_O$ then $C_T = 1000$
else if $P_T == DF_{FR}$ then $C_T = 1400$
else if $P_T == EF_{V1}$ then $C_T = 1076$
else if $P_T == RW_{TO}$ then $C_T = 1251$
else if $P_T == RT_{V1}$ then $C_T = 1201$
else if $P_T == TC_{Ev}$ then $C_T = 1176$
else if $P_T == S_R$ then $C_T = 1350$
end if
3: status = Executed (The current task status is set to Executed.)
Repeat the actions below until the current task is task P_T (task to be Performed).
4: **repeat**
5: $D_G = generated\_path(D_G, O_B)$ (Generate the directed graph with the current task (C_T) as the starting node and task P_T as the ending node. To do this, we use the python package NetworkX.)
6: print(D_G) (Display the directed graph in the visual interface. To do this, we use the python packages Pyvis and Matplotlib.)
7: $T = next\_task(D_G)$ (Select the new current task in the directed graph D_G. As the graph is directed, the tasks are chronologically linked.)
8: status = swrl(C_T) (The status will be set to Executed if C_T was executed successfully, NotExecuted otherwise. Our SWRL rules can be used to find out the status of a given task.)
9: **if** status == Executed **then**
10: $C_T = T$ (The current task C_T is set to the next task T.)
11: $E_T = pattern\_matcher(C_T, R_O)$ (The cognitive agent uses its complex pattern matching mechanism to execute task C_T and cognitively assist the pilot. It proposes actions by selecting the production rules and modules to be activated to manage the current task.)
12: print(E_T) (Display execution time E_T.)
13: **end if**
14: **until** $T == P_T$
15: buffers = [P_T, O_B, E_T, C_T, D_G, T]
16: **return** Buffers

in the reference ontology and the defined production rules, it executes the task and makes personalized recommendations to the human pilot, thus contributing to his training. The time required to complete the task is also displayed. This iterative process continues until the initial goal is reached. During this procedure, the synthetic pilot shares the various possible choices with the human pilot, thus improving his decision-making capabilities in both normal and critical contexts.

To make the reference ontology directly executable, SWRL rules have been defined. These rules enable the synthetic pilot, for a given task, to determine its execution status based on the conditions and constraints previously defined in the reference ontology, ensuring automated and consistent decision-making in the cognitive system. Figure 5.3 shows an example of a SWRL rule for evaluating a task such as 1205 involved in resolving the critical situation Rejected Takeoff after V1. The rule is designed to validate tasks which have two type 1 constraints and a precondition which is itself a task.

```
Task(?t), canBeExecutedBy(?t,?capability), canExecute(?capability,?t),
Person(?p), isResponsible(?p, ?spt), hasCapability(?p, ?capability),
hasSuperTask(?t, ?spt1), hasExecutionStatus(?spt1, LiveExecution),
hasPreCondition(?t, ?pc1),
hasExecutionStatus(?pc1, Executed),
        hasConstraint(?t, ?const1), hasConstraintType(?const1, ?ctype1),
        equal(?ctype1, 1), hasEvaluationCriteria(?const1, ?ev1),
        hasExactValue(?const1, ?ex_val1), hasActualValue(?ev1, ?ac_val1), equal(?ex_val1, ?ac_val1),
        hasConstraint(?t, ?const2), hasConstraintType(?const2, ?ctype2),
        equal(?ctype2, 1), hasEvaluationCriteria(?const2, ?ev2),
        hasExactValue(?const2, ?ex_val2), hasActualValue(?ev2, ?ac_val2), equal(?ex_val2, ?ac_val2)
-> hasExecutionStatus(?t, Executed)
```

Figure 5.3 – SWRL Rule for Evaluating a Task such as 1205.

The cognitive synthetic pilot was developed using specialized technologies to model, simulate, and visualize pilot decision-making processes. PyACTR, a Python library based on the ACT-R cognitive architecture, simulated pilots' cognitive processes. Owlready2 was used to create, manage, and query the reference ontology in OWL format, while Pellet, an inference engine, enabled logical deductions from axioms and relationships. SWRL (Semantic Web Rule Language) defined logical rules related to task execution. For task representation, NetworkX generated and manipulated directed graphs, with Pyvis providing web-based visualizations and Matplotlib supporting 3D rendering. Protégé, an ontology editor, facilitated the development and management of the reference ontology, and UML (Unified Modeling Language) enabled visual representation of concepts and relationships. Figure 5.4 provides a graphical representation of the reference ontology using

the Protégé ontology editor, focusing on task 1076, which is a subtask of the Engine Failure after V1 supertask.

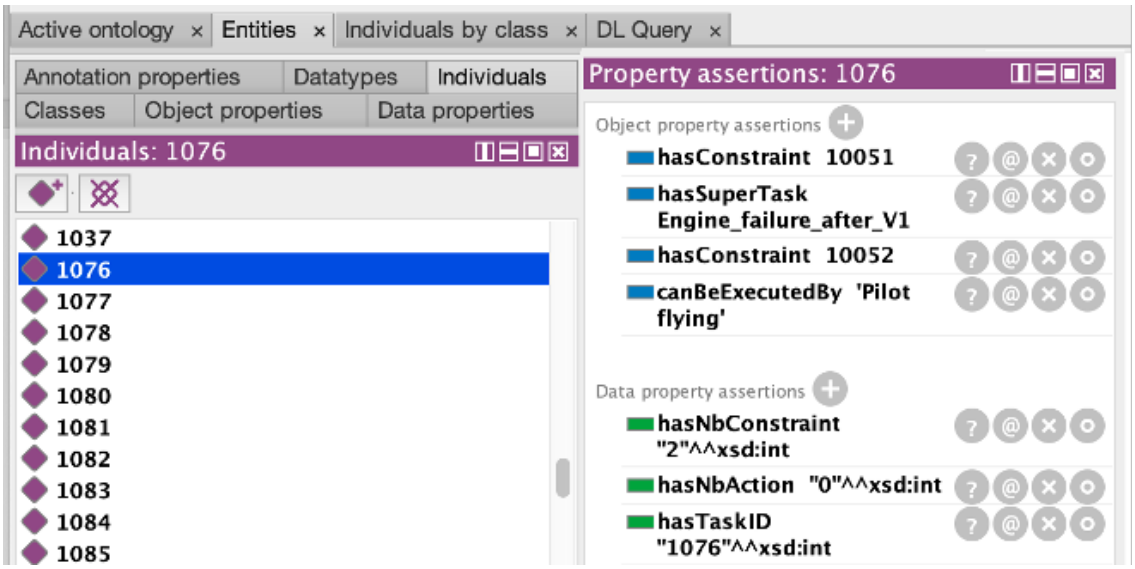


Figure 5.4 – Graphical Representation of Task 1076 from the Reference Ontology.

5.2 Results

To evaluate our model, we conducted several series of tests. We had the synthetic pilot perform several normal takeoffs as well as takeoffs involving various incidents described earlier, such as Engine Failure After V1, Rejected Takeoff after V1, Reactive Windshear at Takeoff, TCAS Event, Dual engine failure with fuel remaining, and Stall recovery. For each scenario, we executed the previously described Algorithm 1 with the synthetic pilot. In each case, we compared the results obtained by the virtual agent with the expected results, based on the procedures established by aviation experts. Figure 5.5 shows a portion of the state-transition diagram of the reference procedure for handling the abnormal situation Dual engine failure with fuel remaining. The complete procedure, comprising over 200 complex states, is too large to present in full, so we have only shown the initial and final parts of the diagram, as is the case for other procedures.

We then compared the tasks dynamically generated by the directed graph produced by the synthetic pilot with those expected at each stage, taken from the reference ontology, which integrates expert knowledge of the domain. Figure 5.6 shows part of the directed graph generated by the synthetic pilot to solve the Dual engine failure with fuel remaining situation, while Figure 5.7 shows the preconditions and constraints required to solve the first task (1400) and the last task (Dual_engine_failure_with_fuel_remaining) of the

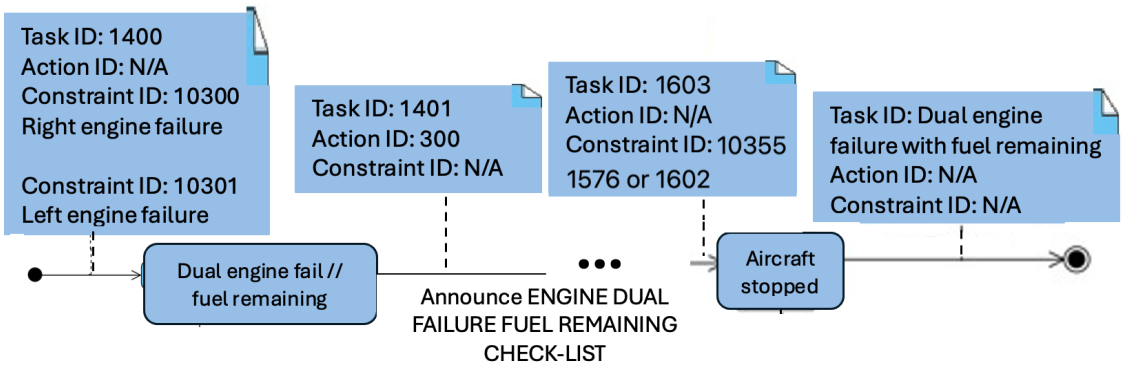


Figure 5.5 – An Extract from the Reference State-Transition Diagram of the Dual Engine Failure with Fuel Remaining Resolution Procedure.

reference ontology associated with the same procedure.

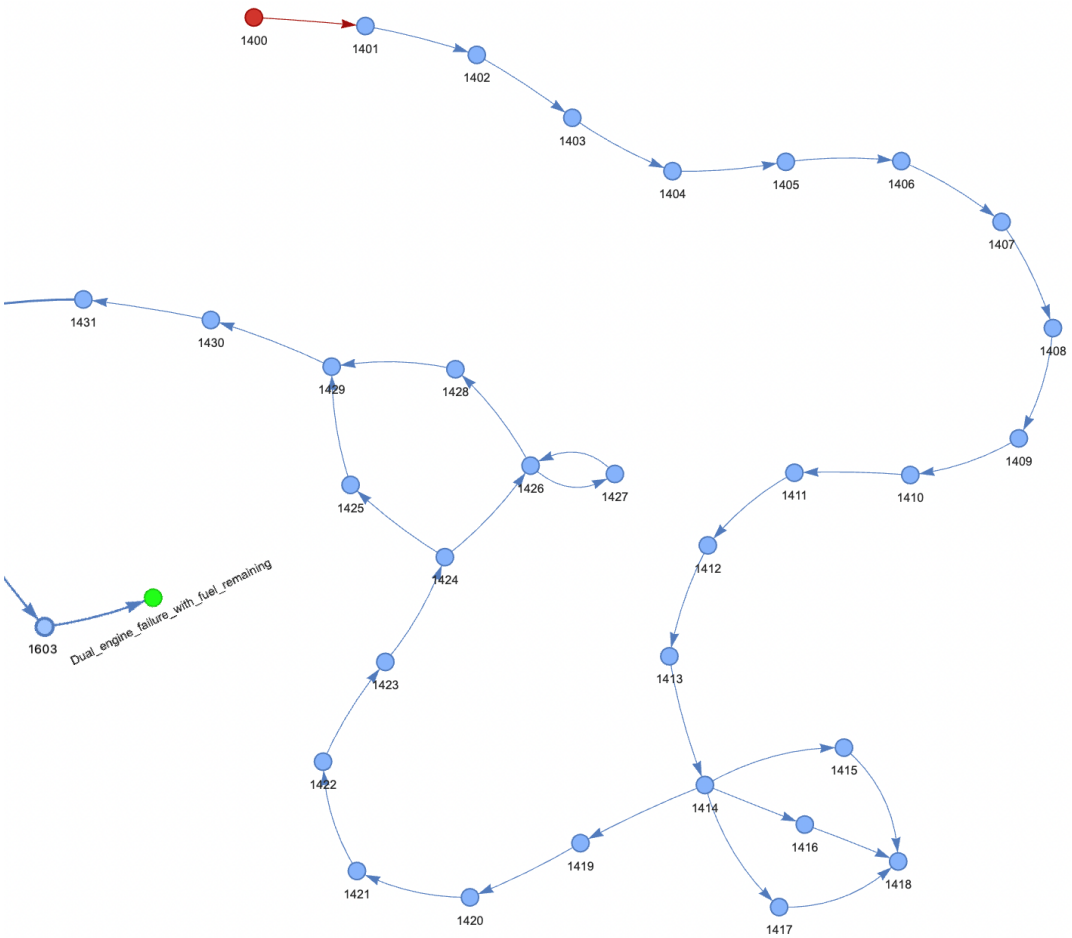


Figure 5.6 – An Extract from the Directed Graph of the Dual Engine Failure with Fuel Remaining Resolution Procedure.

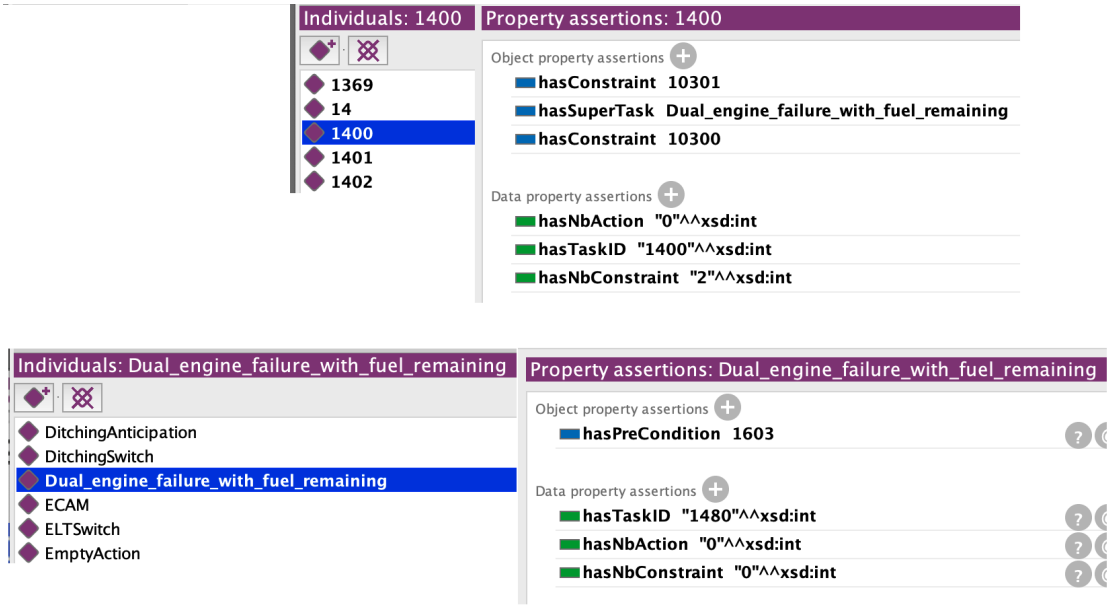


Figure 5.7 – Preconditions and Constraints for the First Task (1400) and the Last Task of the Reference Ontology (Dual_engine_failure_with_fuel_remaining).

To rigorously validate the results produced by the synthetic pilot, we assessed the compliance between the tasks dynamically generated in the directed graph (G_1) (Figure 5.6) and the reference procedures established by experts, recorded in the reference state-transition diagram (G_2) (Figure 5.5). For this purpose, we used the Graph Edit Distance (GED) as a comparison metric. GED measures the minimal cost of a sequence of operations (insertion, deletion, or substitution of nodes or edges) required to transform one graph into another. This approach enables precise quantification of structural differences between graphs (Serratosa, 2021). This metric evaluates the following key aspects :

— Node Matching : Verifying that all tasks (nodes) from the reference state-transition diagram are present in the synthetic pilot's graph. For example, in the Dual Engine Failure with Fuel Remaining scenario (Figure 5.5), we confirmed that critical tasks such as « 1400 » (initial task) and « Dual_engine_failure_with_fuel_remaining » (final task) were included in the generated graph (Figures 5.5 and 5.6).

— Arc Consistency : Validating that the transitions between tasks in the generated graph conform to the procedural logic defined in the reference ontology. This includes verifying sequential and conditional dependencies between tasks, such as the preconditions and constraints illustrated in Figure 5.7. A generated transition is therefore considered valid if and only if it complies with the logical constraints and preconditions defined in the reference ontology.

The GED is defined as :

$$GED(G_1, G_2) = \min_{(e_1,...,e_k) \in T(G_1,G_2)} \sum_{i=1}^{k} c(e_i) \tag{5.1}$$

Where $T(G_1, G_2)$ denotes the set of edit paths transforming G_1 into G_2, and $c(e) \geq 0$ is the cost of each graph edit operation e (vertex insertion, vertex deletion, vertex substitution, edge insertion, edge deletion, edge substitution, etc.).

By achieving 100% node matching and arc consistency across all tested scenarios ($GED(G_1, G_2) = 0$, as no operations were required and $c(e) = 0$), we confirmed that the graphs generated by the synthetic pilot faithfully reproduced the structure and logic of the reference state-transition diagrams defined by piloting standards.

The results obtained enabled us to validate the synthetic pilot's ability to handle takeoffs in both normal and critical situations, in line with expert guidelines. Additionally, the directed task graph generated at each stage by the synthetic pilot reveals a remarkable capacity for adaptation. Indeed, for certain tasks where the reference model had not foreseen an exit situation (dead ends), our synthetic pilot was able to exploit the transitions between tasks in the graph to find solutions deemed appropriate.

Although the reference ontology has already been validated by aviation experts, a further study with experienced pilots is required. This will confirm the relevance and reliability of the results obtained by the cognitive agent, thereby strengthening the validity of our approach.

The synthetic pilot enhances personalized pilot training and decision-making through two distinct modes : autonomous and interactive modes.

In autonomous mode, the cognitive agent independently executes a selected procedure, displaying aircraft environmental data and relevant parameters such as constraint evaluation criteria, action parameters, and preconditions. This enables pilots to observe and understand complete takeoff procedures, including both normal and critical scenarios, reinforcing their learning and decision-making skills. Figure 5.8 illustrates an example.

```
(3.55, 'PROCEDURAL', 'RULE SELECTED: retrieve: ValeurParametreAction')
(3.6, 'PROCEDURAL', 'RULE FIRED: retrieve: ValeurParametreAction')
(3.6, 'g', 'MODIFIED')
(3.6, 'retrieval', 'START RETRIEVAL')
(3.6, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(3.6, 'PROCEDURAL', 'NO RULE FOUND')
(3.65, 'retrieval', 'CLEARED')
(3.65, 'retrieval', 'RETRIEVED: word(cat= ValeurParametreAction, form= Aucune)')
(3.65, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(3.65, 'PROCEDURAL', 'RULE SELECTED: scan: word')
(3.7, 'PROCEDURAL', 'RULE FIRED: scan: word')
(3.7, 'g', 'MODIFIED')
(3.7, 'imaginal', 'MODIFIED')
(3.7, 'retrieval', 'CLEARED')
(3.7, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(3.7, 'PROCEDURAL', 'RULE SELECTED: done')
(3.75, 'PROCEDURAL', 'RULE FIRED: done')
parsed_word Aucune
(3.75, 'g', 'EXECUTED')
(3.75, 'g', 'MODIFIED')
(3.75, 'imaginal', 'CLEARED')
```

Figure 5.8 – Autonomous Mode.

In interactive mode, the synthetic pilot guides the human pilot step by step, requesting expected values, displaying reference ranges, and providing real-time recommendations. If deviations occur, the synthetic pilot assists in corrective actions, acting as an intelligent coaching system. This interactive support helps pilots refine their execution of procedures and improves their ability to respond effectively in complex situations. An example of execution is shown in Figure 5.9.

The two modes of operation (stand-alone and interactive) enable pilots to receive personalized training tailored to their specific needs and skill levels. This dual approach to personalized training not only enhances learning efficiency, but also prepares pilots to face a variety of situations with confidence and competence, thus contributing to overall aviation safety.

```
(1.25, After Actions : True
(1.25, Current constraint : 10001
(1.25, Waiting Dataref Value : CrossWind = >? 5
(1.9, Constraint : 10001  c_type == [2]
(1.95, value ==  5.0
(1.95, minValue ==  0.0
(1.95, maxValue ==  20.0
(1.95, Constraint 10001 => hasConstraintPermission : ConstraintOk
(1.95, Current constraint : 10021
(1.95, Waiting Dataref Value : TailWind = >? 100
(2.15, Constraint : 10021  c_type == [1]
(2.2, value ==  100.0
(2.2, exact_value ==  0.0
(2.2, Waiting Input Dataref c_type 0 | 1 : TailWind = >? 0
(3.35, Constraint 10021 => hasConstraintPermission : ConstraintOk
(3.4, After Constraint : True
(3.4, After SuperTask : True
(3.4, Task 1003 => hasExecutionStatus : Executed
(3.4, After Precondition : True
(3.45, Current action : 4
(3.45, actionParameter : BrakeLeft  , actionValue: 0.0
```

Figure 5.9 – Interactive Mode.

6 Conclusion and Future Works

This study addresses the research questions mentioned in the introduction, and validates the hypotheses formulated by presenting a cognitive assistance approach using ontologies to improve pilot training and decision-making processes. By combining a domain ontology describing the aircraft's external environment and a task ontology describing flight procedures, we have developed a cognitive synthetic pilot based on the ACT-R cognitive architecture. The proposed approach demonstrates the potential of ontologies and cognitive computing to enhance pilots' capabilities and reinforce aviation safety. It helps reduce the risk of human error, while promoting continuous, personalized learning of optimal flight procedures. By providing a formal, shared representation of aeronautical domain knowledge, the reference ontology enables the synthetic pilot to reason and provide relevant support both during normal takeoffs and in abnormal situations occurring during takeoff. The two modes of operation of the synthetic pilot, autonomous and interactive, offer flexibility in personalized training and decision support.

Nevertheless, we must acknowledge certain limitations of our approach. The synthetic pilot relies exclusively on the expert knowledge formalized in the reference ontology, as well as on ACT-R's built-in learning mechanisms, such as reinforcement learning and statistical learning. This dependence could restrict the system's ability to adapt to situations not foreseen by the ontology, or to capture the full complexity of human behavior in dynamic and varied flight contexts.

Future work will aim to extend the tests to realistic flight simulators, such as X-Plane, and to specific aircraft models, like the Airbus A320, to evaluate the robustness and flexibility of the system. Furthermore, although the reference ontology has been validated by aviation experts, it will be essential to consolidate the findings of this study by evaluating the system with real users, particularly pilots and domain experts. An important area for improvement involves integrating machine learning techniques to enhance the synthetic pilot's knowledge and enable it to learn from real flight data, with the aim of accurately replicating human pilot behavior. Finally, it will be crucial to conduct an in-depth study in close collaboration with pilots and civil aviation authorities to address the ethical and regulatory considerations associated with the use of the training platform.

Acknowledgments. This work was supported by the NSERC Alliance program [grant number ALLRP 549083-19]. We are thankful to CRIAQ, Bombardier, CAE, and BMU for their financial support.

5.3 Discussion et Conclusion

Dans ce chapitre, nous avons étendu les capacités du pilote synthétique cognitif en y intégrant la gestion des situations urgentes et anormales pouvant survenir lors du décollage. L'agent cognitif ainsi obtenu joue un rôle d'assistant au pilotage, avec pour objectif d'améliorer la formation des pilotes et de renforcer leur prise de décision en temps réel, aussi bien en conditions normales qu'en situations critiques.

L'un des principaux atouts de cette approche réside dans sa capacité à structurer les connaissances aéronautiques, c'est-à-dire les procédures de pilotage d'un avion définies selon les standards du domaine, et à automatiser le raisonnement du pilote synthétique à l'aide d'un moteur d'inférence et d'un ensemble de règles SWRL. Grâce à cette intégration, le pilote synthétique peut analyser des situations de vol en temps réel, détecter d'éventuelles déviations par rapport aux procédures standardisées, et assister le pilote humain en lui fournissant des recommandations adaptées pendant la phase de décollage, aussi bien en situation normale qu'en situation critique.

L'implémentation du système a reposé sur des outils technologiques spécialisés, notamment :

- PyACTR, une bibliothèque Python pour la modélisation des processus cognitifs de l'architecture ACT-R ;
- Owlready2, une bibliothèque Python pour la gestion et l'exploitation des ontologies en OWL ;
- Pellet, un moteur pour le raisonnement automatique sur les connaissances modélisées ;
- NetworkX et Pyvis, deux bibliothèques Python pour la représentation dynamique des procédures de vol sous forme de graphes orientés ;
- SWRL, un langage de formalisation des règles sémantiques pour automatiser les décisions du pilote synthétique.

L'utilisation combinée de ces technologies a permis au pilote synthétique de gérer des situations imprévues, en identifiant des solutions alternatives lorsque l'ontologie de référence ne prévoyait pas de sortie spécifique. Cette flexibilité a renforcé son utilité en contexte opérationnel, en permettant une adaptation dynamique aux scénarios complexes ou inattendus.

Le pilote synthétique est capable d'assister efficacement les pilotes selon deux modes de fonctionnement :

- Le mode autonome, où il exécute les procédures et affiche les résultats en temps réel sur un terminal ;

— Le mode interactif, où il agit comme un coach cognitif, en guidant le pilote humain et en lui proposant des actions correctives adaptées au contexte.

L'approche développée dans ce chapitre indique que le coaching cognitif fourni par le pilote synthétique peut améliorer la formation des pilotes et leur prise de décision (Zhao *et al.*, 2024; Kozak, 2019), en leur permettant d'acquérir plus rapidement les compétences nécessaires à la gestion des décollages, en procédures normales et anormales, contribuant ainsi à l'amélioration de la sécurité aérienne.

Cependant, Plusieurs axes d'amélioration restent à explorer. Tout d'abord, la validation des actions du pilote synthétique dans un environnement de simulation plus réaliste devra être conduite, idéalement en collaboration avec des pilotes professionnels. Par ailleurs, l'intégration de techniques d'apprentissage automatique permettrait d'enrichir les capacités du pilote synthétique, en exploitant les données comportementales des pilotes issues de vols réels, et en diversifiant les actions qu'il est en mesure de simuler. Enfin, une attention particulière devra être portée aux questions éthiques soulevées par l'utilisation d'une telle plateforme dans un contexte de formation. Le chapitre suivant, intitulé *Discussion Générale*, dressera un bilan des contributions apportées dans cette thèse. Il analysera de manière critique les résultats obtenus, mettra en lumière les limites de l'approche proposée, et explorera les perspectives de recherche pour le perfectionnement futur du pilote synthétique cognitif.

CHAPITRE 6
DISCUSSION GÉNÉRALE

6.1 Introduction

Ce chapitre présente une analyse intégrative des résultats de notre recherche, en offrant une vue d'ensemble du développement et de l'évaluation du pilote synthétique fondé sur l'architecture cognitive ACT-R. Ce dernier s'appuie sur l'ontologie de référence des procédures de pilotage pour structurer et exploiter la connaissance experte formalisée par les spécialistes du domaine, assurant ainsi un raisonnement conforme aux standards opérationnels en matière de pilotage.

Notre travail s'inscrit dans le cadre d'un projet plus vaste appelé Pilot-AI, mené de 2020 à 2024. Ce projet résulte d'une collaboration entre plusieurs institutions académiques, dont l'UQAM et l'UdeM, des partenaires industriels, parmi lesquels Bombardier, BMU et CAE, ainsi que des organismes de recherche, dont le CRSNG et le CRIAQ. Cette initiative avait pour objectif de renforcer l'assistance cognitive des pilotes en tirant parti des avancées en intelligence artificielle et en modélisation cognitive. Ainsi, nous commencerons par présenter le cadre expérimental dans lequel cette recherche a été menée. Nous examinerons ensuite la signification de nos résultats en évaluant leur impact tant sur le plan informatique que cognitif. Par la suite, nous identifierons les contraintes et défis rencontrés à travers une analyse comparative des différentes initiatives menées au sein du projet Pilot-AI et leur intégration dans notre travail. Enfin, nous proposerons des perspectives de recherche pour approfondir et améliorer ce travail.

Il convient de rappeler que, dans le cadre de cette thèse, le pilote synthétique devait se concentrer exclusivement sur la connaissance experte formalisée par l'ontologie de référence afin de reproduire les comportements d'un pilote humain, tout en s'appuyant de manière implicite sur les mécanismes d'apprentissage natifs de l'architecture ACT-R, tels que l'apprentissage par renforcement et l'apprentissage statistique.

6.2 Présentation du contexte expérimental et des instruments de mesure

Dans le cadre du projet Pilot-AI, une expérience utilisant le simulateur de vol X-Plane avec un Airbus A320 a été réalisée pour recueillir différentes données sur les pilotes en temps réel lors d'une procédure de décollage, en conditions normales et anormales, sur la piste 06L de l'aéroport de Montréal-Trudeau (YUL).

Les données collectées incluaient la charge de travail, la dilatation de la pupille et la fréquence cardiaque. Puisque cette expérience impliquait la participation de pilotes humains, un certificat d'éthique, portant le numéro CERSES-3849, a été obtenu auprès du comité d'éthique de l'Université de Montréal.

Les participants ont été recrutés chez Bombardier et CAE, tous travaillant dans un domaine lié à l'industrie aéronautique, sans qu'une licence de pilote ne soit obligatoire. Au total, 13 participants ont complété l'expérience, réalisant 136 décollages. Tous étaient des hommes. Parmi eux, 7 étaient pilotes, dont 5 avaient de l'expérience sur l'airbus A320.

L'expérience s'est déroulée au laboratoire de l'UdeM, dans la salle 3332 du Pavillon André-Aisenstadt. L'environnement a été conçu pour minimiser les distractions et maximiser l'immersion. Les participants étaient installés devant un simulateur X-Plane, équipé d'un joystick, de manettes des gaz et de pédales similaires à ceux d'un A320 (voir Figure 6.1).

Les dispositifs de commande de vol et de mesure utilisés incluaient :

- Thrustmaster TCA Sidestick Airbus Edition (PC) : Joystick reproduisant le manche de commande (sidestick) utilisé dans les avions Airbus, permettant de contrôler les mouvements de l'avion (roulis et tangage) de manière précise et réaliste ;
- Thrustmaster TCA Quadrant Airbus Edition (PC) : Quadrant reproduisant les leviers de commande des gaz (throttle) des avions Airbus, permettant de gérer la puissance des moteurs en ajustant la poussée, et simulant ainsi les phases de décollage, de croisière et d'atterrissage ;
- Pédales de gouvernail TFRP : Pédales servant à contrôler le gouvernail de direction (palonnier) et les freins, permettant de gérer le lacet (mouvement latéral) et le freinage ;
- Polar H10 : Capteur de fréquence cardiaque permettant de surveiller et de mesurer les battements du cœur en temps réel ;
- Gazepoint GP3 : Dispositif de suivi oculaire (eye tracking) permettant de suivre les mouvements des yeux et la direction du regard, ainsi que de mesurer la dilatation pupillaire ;
- Casque EEG NCO : Dispositif permettant de mesurer l'activité cérébrale par électroencéphalographie (EEG).

L'annexe A présente les équipements de navigation utilisés, tandis que l'annexe B détaille les équipements de mesure. Les annexes C et D fournissent une vue élargie de la figure 6.1. L'annexe C présente une ex-

périmentation impliquant un pilote, un copilote ainsi que des participants, tandis que l'annexe D illustre l'environnement du poste de pilotage utilisé pour les tests.

Figure 6.1 – À gauche, l'environnement de simulation ; à droite, une expérimentation.

Avant l'expérience, les participants recevaient un document sur les procédures de décollage de l'A320. Après une séance de familiarisation de 30 minutes, les sessions de décollage étaient lancées, avec calibration des outils avant chaque vol.

La Figure 6.2 présente l'architecture de l'environnement de simulation.

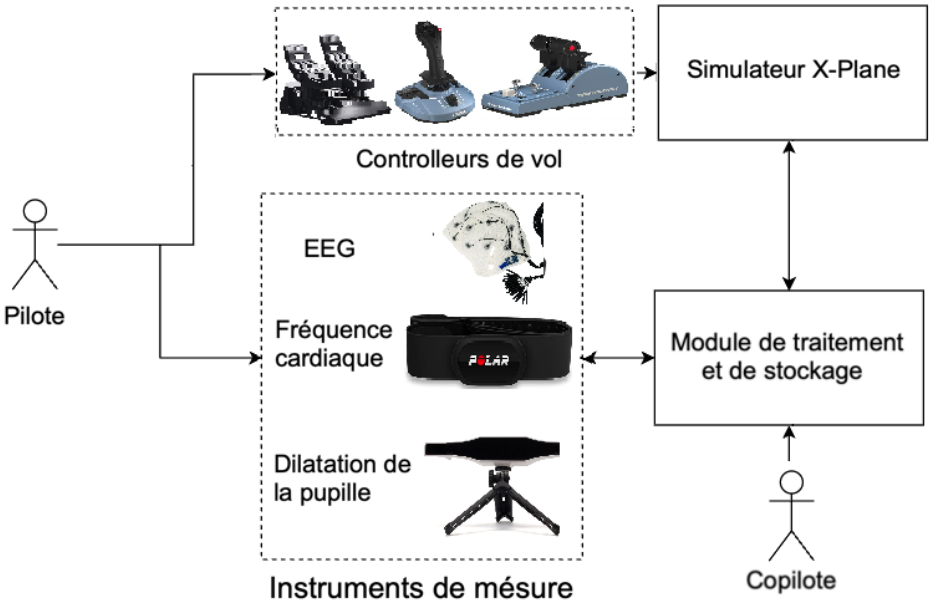


Figure 6.2 – Architecture de l'environnement de simulation.

6.3 Contributions de la thèse

Cette thèse présente des avancées significatives, tant sur le plan informatique que cognitif, reflétant la nature interdisciplinaire de notre recherche. Notre contribution principale réside dans l'association d'un modèle formel de connaissances (l'ontologie de référence) et d'une architecture cognitive (ACT-R), en vue de créer un agent cognitif capable de reproduire les processus mentaux d'un pilote humain.

6.3.1 Contributions sur le plan informatique

Cette thèse a apporté plusieurs contributions significatives sur le plan informatique. Elle a permis d'utiliser des méthodes algorithmiques et informatiques, notamment les techniques avancées de représentation des connaissances, de raisonnement automatique et de théorie des graphes, pour implémenter l'agent cognitif.

Plus précisément, cette thèse a abouti au développement d'un agent cognitif capable de simuler le comportement d'un pilote humain en situation de vol. Ce développement repose sur l'architecture cognitive ACT-R, mise en œuvre à l'aide de la bibliothèque Python PyACT-R, afin de modéliser les processus mentaux impliqués dans la perception, l'attention, la mémoire, la prise de décision et l'exécution d'actions, que ce soit dans des contextes de décollage standards ou critiques.

Afin d'enrichir les capacités de raisonnement de cet agent et de les adapter aux spécificités du domaine aéronautique, nous avons d'abord utilisé le modèle de référence ontologique élaboré par Courtemanche *et al.* (2022), puis nous l'avons adapté à notre contexte (Tamkodjou Tchio *et al.*, 2025). Cette adaptation a conduit à l'élaboration de l'ontologie de référence, qui formalise de manière explicite les connaissances liées aux tâches de pilotage et à leur environnement opérationnel, constituant ainsi le socle informationnel sur lequel s'appuie l'agent cognitif pour orienter ses décisions et ses actions.

Par ailleurs, notre approche a permis d'intégrer des règles SWRL que nous avons créées pour enrichir les capacités de raisonnement de l'agent cognitif. Ces règles ont permis d'assurer une exécution automatique et contextuelle des tâches en fonction des conditions de vol et des contraintes définies dans l'ontologie, facilitant ainsi l'adaptation du système à divers scénarios de vol.

Enfin, l'agent cognitif génère un graphe orienté de tâches à partir des connaissances extraites de l'ontologie. Ce graphe permet de représenter dynamiquement l'enchaînement des tâches en fonction de la situation

de vol, tout en facilitant la planification, l'organisation, la visualisation et la prise de décision.

6.3.2 Contributions sur le plan cognitif

Sur le plan cognitif, cette thèse a permis d'explorer la modélisation des processus mentaux des pilotes humains, en se basant sur l'architecture cognitive ACT-R. L'agent cognitif développé reproduit fidèlement le cycle cognitif d'un pilote humain lors de l'exécution d'une tâche de pilotage, en simulant les processus de perception, de mémorisation et de prise de décision.

Par ailleurs, nous avons mis en évidence la capacité de l'agent à détecter et à corriger les erreurs potentielles grâce à l'intégration d'un modèle de coaching cognitif. Cette approche permet d'améliorer l'apprentissage et la formation des pilotes en leur fournissant une assistance adaptative et contextuelle.

Enfin, la mise en place d'un modèle de mémoire déclarative et procédurale inspiré des principes d'ACT-R renforce la compréhension du fonctionnement cognitif humain dans des situations complexes comme le pilotage d'un avion.

6.4 Limites de la recherche et piste de solutions explorées

Bien que cette recherche ait apporté des contributions significatives dans le domaine de l'aviation, elle présente néanmoins certaines limites qu'il convient de reconnaître. Dans la suite de cette section, nous examinerons en détail ces limites et proposerons des pistes de solutions préliminaires qui pourraient être explorées dans le cadre de recherches futures, afin de les surmonter et d'enrichir les travaux amorcés par cette étude.

6.4.1 Tests en simulateurs de vol certifiés

La validation expérimentale du pilote synthétique s'est principalement appuyée sur des simulations informatiques et des tests en environnement contrôlé. Si ces expériences ont permis d'évaluer ses performances dans divers scénarios, elles ne sauraient se substituer à une évaluation en conditions réelles ou sur simulateur de vol certifié. À cet effet, des travaux préliminaires ont déjà été engagés à l'aide de la plateforme X-Plane, ouvrant des perspectives prometteuses pour le déploiement futur du pilote synthétique.

Le simulateur X-Plane (pour "eXperimental-Plane") constitue une gamme complète de moteurs de simulation aéronautique, conçue et commercialisée par l'entreprise Laminar Research depuis son lancement initial en 1995. Cette plateforme de simulation se distingue par son adoption généralisée auprès d'organisations prestigieuses, notamment la NASA, ainsi que de grands constructeurs aéronautiques comme Boeing, Cirrus, Cessna et Piper, sans oublier des compagnies aériennes internationales telles que Northwest ou Japan Airlines (Baomar et Bentley, 2016; Laminar Research, 2017). L'un des principaux atouts de la plateforme X-Plane réside dans son architecture ouverte, qui permet une communication bidirectionnelle avec des applications externes. Cette fonctionnalité facilite l'échange en temps réel de données relatives aux paramètres de vol et aux commandes de pilotage, ce qui en fait un environnement idéal pour l'intégration et le test de systèmes comme le pilote synthétique (Bodeen, 2011).

Selon le Laminar Research (2017), la solution offre nativement un vaste catalogue d'aéronefs civils et militaires, accompagné d'une cartographie mondiale détaillée couvrant la quasi-totalité du globe terrestre. La flexibilité du système est considérablement renforcée par son architecture modulaire basée sur des plugins, permettant aux utilisateurs d'élargir les fonctionnalités du logiciel via le développement de modules personnalisés. Cette extensibilité favorise la création d'environnements virtuels originaux ou la reproduction fidèle de zones géographiques spécifiques. La modularité de X-Plane en fait également un outil particulièrement adapté pour la recherche et la formation professionnelle.

X-Plane est largement utilisé dans la recherche aéronautique et la formation des pilotes. Par exemple, une étude récente a vérifié la crédibilité des données de vol simulées par X-Plane en les comparant avec des données réelles et calculées (Yokoyama et Harada, 2021).

La version actuelle est X-Plane 12, lancée officiellement en septembre 2022, mais dans le cadre de cette étude, nous avons déployé X-Plane 11, configuré pour assurer une transmission bidirectionnelle des données pertinentes via des paquets réseau, avec une fréquence d'échantillonnage de 10 Hz, soit toutes les 0,1 seconde.

Pour supporter la communication avec l'environnement de simulation X-Plane, nous avons utilisé l'API décrite dans (Pietracupa *et al.*, 2023). Cette interface de programmation, conçue en C++, offre des fonctionnalités permettant à la fois de récupérer et de modifier les données du simulateur, servant ainsi de pont pour interagir dynamiquement avec X-Plane.

L'organisation générale de l'API du simulateur et de ses interactions est schématisée à la figure 6.3.

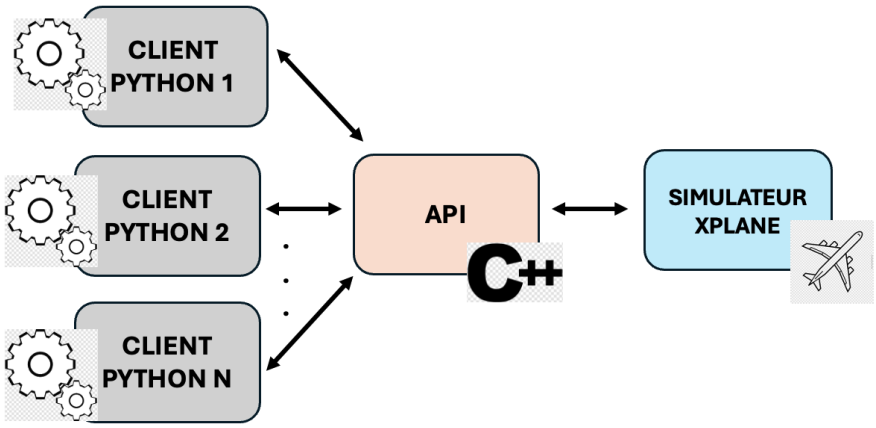


Figure 6.3 – Architecture de haut niveau de l'API du simulateur.

Les fonctions principales utilisées pour communiquer avec XPlane sont `getDataRef` et `setDataRef`, dont la description est donnée ci-dessous :

- La fonction getDataRef() permet l'extraction de la valeur actuelle d'une référence spécifique depuis l'environnement de simulation. Pour récupérer une valeur, il suffit de fournir l'identifiant de la référence du paramètre souhaité. En retour, la fonction fournit la valeur en temps réel de ce paramètre. L'utilisation de cette fonction est particulièrement utile pour le pilote synthétique car elle lui permet de récupérer et d'évaluer les contraintes environnementales liées à l'aéronef.
- La fonction setDataRef() offre au client la possibilité de modifier directement la valeur d'un paramètre (dataref) au sein de l'environnement de simulation. La capacité à ajuster ces valeurs présente un avantage considérable puisqu'elle permet d'exécuter les actions du pilote synthétique dans l'environnement X-Plane.

L'algorithme ci-dessous décrit le processus par lequel le pilote synthétique interagit avec le simulateur X-Plane à l'aide des primitives de communication getDataRef() et setDataRef(). À l'aide du polymorphisme, le pilote synthétique utilise directement les fonctions getDataRef() et setDataRef(). Les fonctions et variables utilisées dans l'algorithme sont définies comme suit :

- infos : Valeur ou plage de valeurs d'un paramètre donné, obtenue à partir de l'ontologie de référence.
- setDataRef(X) : Fonction envoyant la valeur du paramètre X au simulateur X-Plane pour mise à jour.
- getDataRef(X) : Fonction récupérant la valeur du paramètre X de l'environnement d'exécution du

Algorithme : Communication entre le pilote synthétique et X-Plane

Entrée : Ontologie de référence

Paramètre : gRef

Sortie : sRef, buffers

1: **while** true **do**

2: Récupérer la valeur actuelle du paramètre gRef depuis l'environnement X-Plane et la stocker dans gRef.

$gRef = getDataRef(gRef)$

3: Consulter l'ontologie de référence pour obtenir des informations sur gRef.

$infos = search\_information\_in\_referenceModel(gRef)$

4: Exécuter la règle issue de la mémoire procédurale correspondant à infos et retourner le paramètre sRef à envoyer à X-Plane.

$sRef = pattern\_matcher(infos)$

5: Utiliser le dictionnaire ontologique pour obtenir la valeur correspondante de sRef dans la nomenclature de X-Plane.

$sRef = processing(sRef)$

6: Envoyer la valeur sRef au simulateur X-Plane pour mise à jour.

$setDataRef(sRef)$

7: Mettre à jour la mémoire du pilote synthétique avec la nouvelle valeur de gRef.

$buffers(gRef)$

8: **end while**

9: Stocker les tampons mis à jour.

$buffers = [sRef, gRef]$

10: Retourner les tampons mis à jour

return $buffers$

simulateur X-Plane.

— processing(X) : Fonction qui utilise le dictionnaire ontologique pour récupérer la valeur correspondante dans X-Plane à la valeur X de l'ontologie.

— buffers(X) : Met à jour les tampons en modifiant la valeur du paramètre X dans la mémoire du pilote synthétique.

Afin de garantir la compatibilité du système avec différents simulateurs, l'API est configurée à l'aide d'un fichier `Subscriptions.yaml`. Ce fichier contient des étiquettes associées à des labels, où chaque étiquette correspond à une chaîne de référence spécifique à un simulateur, représentant un système particulier de l'aéronef. Par exemple, comme illustré dans la figure 6.4, l'étiquette `LeftThrustLever` est utilisée par les clients de l'API pour identifier le système concerné.

En parallèle, la balise `sim/cockpit2/engine/actuators/throttle_ratio[0]` est celle qu'X-Plane utilise pour désigner la manette de poussée du premier moteur. Cette approche permet aux applications interagissant avec l'API de référencer de manière cohérente des éléments comme le levier de poussée, tout en offrant la possibilité de modifier les spécifications du simulateur directement dans les fichiers YAML au besoin.

De plus, la fréquence d'interrogation (en Hz) des étiquettes peut être ajustée en fonction des besoins. Par exemple, une fréquence plus élevée peut être attribuée aux systèmes critiques ou fréquemment sollicités, tels que les pédales ou les leviers latéraux, tandis qu'une fréquence réduite peut être appliquée aux éléments moins utilisés, comme certains boutons. Cette flexibilité permet d'optimiser la charge du système, notamment lorsque de nombreuses étiquettes sont surveillées simultanément, en évitant une sollicitation excessive des ressources.

```
    description: "Sidestick position X from 0 to 1. Use this label to modify cockpit"

SideStickPositionY:
    tag: "sim/joystick/yoke_pitch_ratio"
    frequency: 10
    description: "Sidestick position Y from 0 to 1. Use this label to modify cockpit"

LeftThrustLever:
    tag: "sim/cockpit2/engine/actuators/throttle_ratio[0]"
    frequency: 10
    description: "Engine 1 Thrust value ratio between 0-1"

RightThrustLever:
    tag: "sim/cockpit2/engine/actuators/throttle_ratio[1]"
    frequency: 10
    description: "Engine 2 Thrust value ratio between 0-1"

FMSDirection:
    tag: "sim/cockpit/gyros/psi_ind_degm3"
    frequency: 10
    description: ""

Altitude:
    tag: "sim/flightmodel/misc/h_ind"
    frequency: 10
    description: ""

FlapLeverPosition:
    tag: "sim/flightmodel/controls/flaprqst"
    frequency: 10
    description: "Float between 0 and 1, Requested flap deployment, 0 = off, 1 = max"

FlapLever:
    tag: "a320/Aircraft/Cockpit/Pedestal/FlapsLever"
    frequency: 10
    description: ""
```

Figure 6.4 – Structure et paramétrage du fichier Subscriptions.yaml pour l'API X-Plane.

Le Runner est un composant essentiel qui permet au pilote synthétique de collecter des informations et de rester synchronisé avec le simulateur X-Plane. Pour fonctionner, il doit charger le dictionnaire de données de l'ontologie de référence, créer des structures facilitant l'accès aux données pendant la simulation, puis se connecter à l'API du simulateur. Ce dictionnaire joue un rôle crucial en assurant l'alignement et l'interprétation correcte des données entre l'ontologie et X-Plane, les variables de l'ontologie et celles de X-Plane n'ayant ni la même nomenclature ni les mêmes valeurs. Il est donc nécessaire d'établir une correspondance entre les deux. X-Plane utilise une nomenclature d'étiquettes de données spécifique à chaque système d'aéronef pris en charge. En revanche, l'ontologie de référence repose sur une nomenclature unique d'étiquettes de données, qui diffère néanmoins de chacune des nomenclatures de X-Plane. L'annexe E présente un extrait du dictionnaire de correspondance, en tenant compte du fait que l'appareil simulé est un Airbus A320.

La Figure 6.5 illustre le processus d'exécution des tâches par X-Plane, en suivant les instructions issues de la communication entre le pilote synthétique et le Runner. Ce dernier exploite le dictionnaire de données ontologique et interagit avec l'API du simulateur.

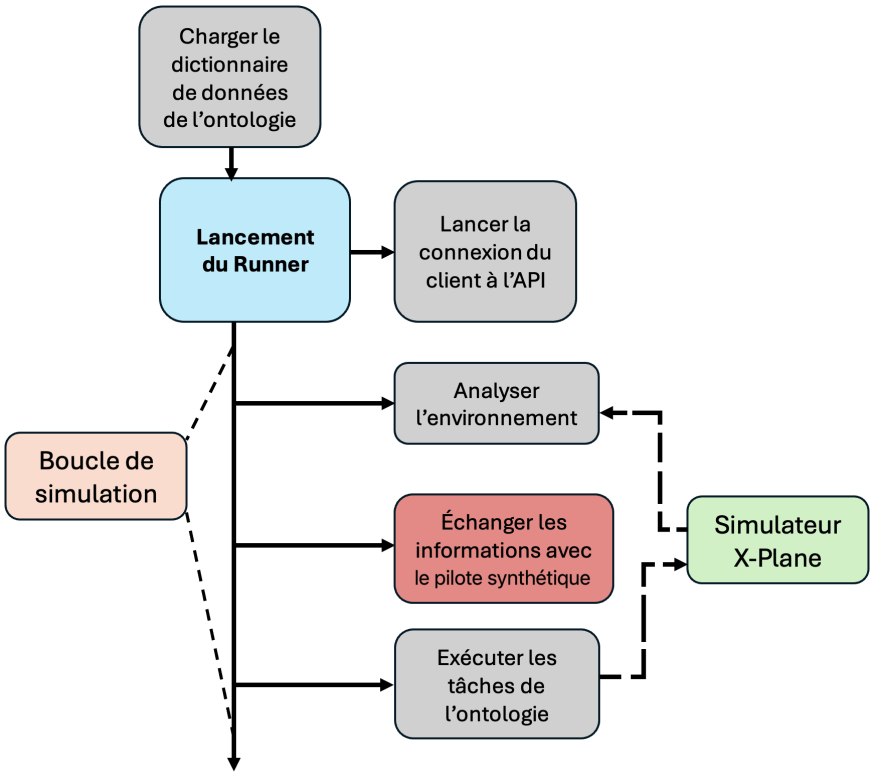


Figure 6.5 – Diagramme d'exécution de X-Plane via les instructions issues de la communication entre le pilote synthétique et le Runner.

La figure 6.6 présente l'architecture de connexion et de communication du pilote synthétique avec X-Plane via le Runner et l'API en utilisant les fonctions prédéfinies getDataRef() et setDataRef().

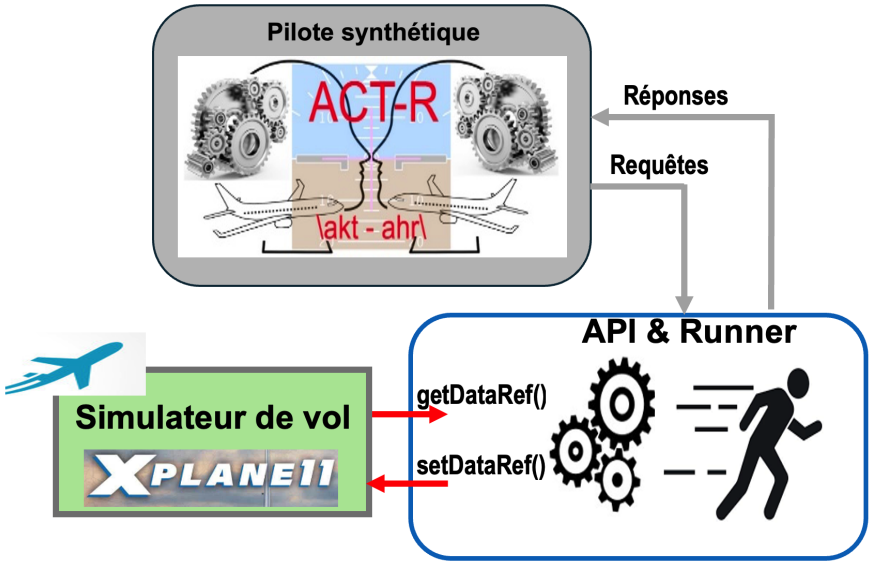


Figure 6.6 – Communication entre le pilote synthétique et X-Plane à travers le Runner et l'API.

La Figure 6.7 représente la chaîne de traitement des instructions à partir du pilote synthétique avant leur exécution dans le simulateur X-Plane.

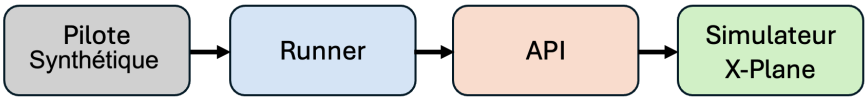


Figure 6.7 – Parcours d'instructions du pilote synthétique jusqu'à X-Plane.

À l'aide d'un code, dont un extrait est présenté à la Figure 6.8, le pilote synthétique a communiqué avec le simulateur X-Plane via l'API et le Runner. En réponse à sa requête (Figure 6.9), il a reçu une confirmation indiquant que l'altitude transmise était de 5000 pieds et la vitesse de 250 nœuds.

```
import time
import shutil
from pathlib import Path
from owlready2 import *
import src.deviation as deviation
import win32gui

# Chargement de l'ontologie de référence par le Pilote virtuel
onto_task = get_ontology("/Users/guycarlostamkodjoutchio/Desktop/ACTR/Ontology/task.owl").load()
onto_domain = get_ontology("/Users/guycarlostamkodjoutchio/Desktop/ACTR/Ontology/domain.owl").load()
print("Ontologies de référence chargées par le Pilote virtuel")

# Configuration de la connexion à X-Plane
XP_IP = "127.0.0.1"
XP_SEND_PORT = 49000  # Port pour envoyer des commandes
XP_RECEIVE_PORT = 49005  # Port pour recevoir des données

# Création du socket UDP pour envoyer des données
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.settimeout(2)  # Timeout pour éviter les blocages
sock.bind((XP_IP, XP_RECEIVE_PORT))  # Liaison pour recevoir les données


from XPlaneApi import XPlaneClient

# Initialisation du client XPlane par le Pilote virtuel
api_client = XPlaneClient("Runner")

# Appel de la commande de connexion
if not api_client.connect():
    print("Connection client 1 failed")
    exit()

print("Connection successful (Pilote virtuel connecté)")

# Vérification de l'état de la connexion
if api_client.isConnected():
    print("Connexion active avec XPlane")
else:
    print("Connexion perdue")
    exit()

# Exemple d'envoi de données avec setDataRef() par le Pilote virtuel
api_client.setDataRef("sim/flightmodel/position/y_agl", 5000)  # Altitude
api_client.setDataRef("sim/flightmodel/position/true_airspeed", 250)  # Vitesse
print("Données envoyées à XPlane")

# Récupération de données de l'environnement avec getDataRef()
altitude = api_client.getDataRef("sim/flightmodel/position/y_agl")
speed = api_client.getDataRef("sim/flightmodel/position/true_airspeed")

print(f"Received Altitude: {altitude} ft (Données reçues par le Pilote virtuel)")
print(f"Received Speed: {speed} knots (Données reçues par le Pilote virtuel)")

# Fermeture des connexions
api_client.disconnect()
sock.close()
print("Client déconnecté et socket fermé")

```

Figure 6.8 – Extrait du code de connexion au Runner et de communication avec X-Plane.

```
Python 3.9.7 (default, Sep 16 2021, 13:09:58)
[GCC 7.5.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
Ontologies de référence chargées par le Pilote virtuel
Connection successful (Pilote virtuel connecté)
Connexion active avec XPlane
Données envoyées à XPlane
Received Altitude: 5000.0 ft (Données reçues par le Pilote virtuel)
Received Speed: 250.0 knots (Données reçues par le Pilote virtuel)
Client déconnecté et socket fermé
>>>
```

Figure 6.9 – Réponse reçue par le pilote synthétique lors d'une communication avec X-Plane via le Runner et l'API.

Les données transmises par le pilote synthétique dans la communication illustrée à la Figure 6.8 ont été générées manuellement. Néanmoins, cette situation démontre que le pilote synthétique parvient à établir une connexion avec X-Plane et à échanger des informations avec le simulateur de vol. Toutefois, ses capacités actuelles ne lui permettent pas encore de réaliser une communication reposant sur une inférence dérivée d'une règle de production issue de l'analyse d'une situation perçue. Des améliorations supplémentaires seront donc requises afin d'intégrer ces fonctionnalités et d'aboutir à une interaction plus autonome et adaptative entre les deux systèmes.

6.4.2 Amélioration du modèle par apprentissage machine

La flexibilité du modèle cognitif demeure limitée par la structure préétablie des règles SWRL et par le moteur d'inférence utilisé. Bien que l'agent cognitif soit en mesure d'adapter son raisonnement en fonction des contraintes opérationnelles et des tâches à accomplir, il ne dispose pas encore d'un mécanisme d'apprentissage automatique, en dehors de celui natif à ACT-R, lui permettant d'évoluer dynamiquement à partir des expériences passées. Cependant, il est envisageable d'étendre la mémoire du pilote synthétique en intégrant des profils comportementaux de pilotes obtenus via l'apprentissage automatique.

Une étude menée dans le cadre du projet Pilot-AI a exploité l'apprentissage automatique afin de classifier les pilotes en fonction de leurs profils comportementaux (Tato *et al.*, 2022, 2023). Les auteurs ont exploré l'utilisation de l'apprentissage automatique pour analyser et modéliser les comportements de pilotage, mettant

ainsi en évidence des profils comportementaux spécifiques aux pilotes d'avions. L'étude a porté principalement sur la phase de décollage d'un Airbus A320 et visait à classifier les pilotes selon leur gain (élevé vs faible) ainsi qu'à prédire leurs actions en temps réel. L'approche repose sur l'analyse des données de vol issues du simulateur CAE et sur l'utilisation de réseaux de neurones récurrents (LSTM) pour modéliser ces comportements. L'objectif final est de développer un pilote synthétique capable d'exécuter les différents modèles appris dans le simulateur X-Plane, fournissant ainsi une validation pratique des modèles développés. Les profils de pilotes sont principalement appris à partir de segments de vol significatifs, plutôt qu'à partir de l'ensemble des données de vol, afin de capturer avec précision les variations comportementales des pilotes tout en évitant la perte d'informations associée à une analyse trop généralisée. Les experts du domaine ont identifié dix segments distincts dans la phase de décollage d'un Airbus A320, dont trois sont considérés comme les plus significatifs pour l'analyse des comportements de pilotage. Ces segments correspondent aux phases critiques où les actions des pilotes influencent directement la sécurité et la performance du vol :

- Segment 5 : Action du pilote sur le manche de tangage en fonction des conditions de vent.
- Segment 6 : Rotation de l'avion avant le décollage.
- Segment 7 : Décollage et montée initiale.

L'étude a permis d'identifier deux principaux profils comportementaux de pilotage, définis en fonction du niveau d'agressivité du pilote dans ses manœuvres :

- Les pilotes à gain élevé (High Gain) qui adoptent une approche caractérisée par des ajustements rapides et fréquents des commandes. Ils exercent un contrôle plus agressif sur le manche et les pédales, ce qui leur permet de réagir rapidement aux changements de situation. Toutefois, cette réactivité accrue peut parfois engendrer des instabilités, notamment lorsque les corrections sont excessives ou mal dosées.
- Les pilotes à gain faible (Low Gain) qui privilégient plutôt une approche progressive et modérée dans leurs actions sur les commandes. Ils effectuent des corrections moins fréquentes et plus douces, favorisant ainsi un vol plus stable et fluide. Cependant, cette prudence peut entraîner une latence dans la réponse aux imprévus, augmentant ainsi le risque de réaction tardive face à des situations critiques.

Les auteurs ont développé un pilote synthétique, testé dans l'environnement de simulation X-Plane, afin de permettre l'exécution automatique du modèle appris et de simuler les actions des pilotes humains en

fonction de leur gain (faible ou élevé) lors de la phase de décollage d'un Airbus A320. La communication avec le simulateur X-Plane est assurée via le Runner, qui encapsule le modèle appris, et l'API décrits dans la section précédente. Cependant, ce pilote synthétique est encore en phase de perfectionnement.

À l'instar des expérimentations menées avec X-Plane, le pilote synthétique basé sur le modèle appris et celui fondé sur l'architecture cognitive ACT-R ont pu établir une communication via l'API et échanger des données simulées. Bien que ces échanges aient été réalisés à partir de données générées manuellement, cette expérimentation démontre la capacité de notre pilote synthétique à se connecter au pilote synthétique fondé sur le modèle appris et à échanger efficacement des informations. Cependant, les capacités actuelles du pilote synthétique ne lui permettent pas encore d'exploiter pleinement les connaissances issues du modèle appris, ni de les appliquer au pilotage d'un avion dans des scénarios réels, que ce soit pour un décollage en conditions normales ou dans des situations critiques. Des développements supplémentaires seront nécessaires pour intégrer ces fonctionnalités et permettre une interaction plus dynamique et autonome entre les deux systèmes.

6.4.3 Intégration des données sur la charge mentale et sur l'attention des pilotes

Le pilote synthétique, fondé sur l'architecture cognitive ACT-R et s'appuyant sur l'ontologie de référence pour structurer ses connaissances, a déjà démontré qu'il pouvait exécuter des tâches de pilotage, interagir avec l'environnement de simulation X-Plane, ainsi qu'avec le pilote synthétique fondé sur le modèle appris. Ce dernier est capable de réagir aux conditions spécifiques du vol et d'adapter ses actions en fonction des situations rencontrées. Cependant, une dimension essentielle de la cognition humaine reste encore insuffisamment prise en compte dans ce système : la charge mentale et l'attention du pilote.

En effet, toujours dans le cadre du projet Pilot-AI, deux équipes de chercheurs ont étudié la charge mentale et l'attention des pilotes à travers l'expérience décrite au début de ce chapitre, à la section 6.2.

L'étude menée par la première équipe avait pour objectif principal de mesurer et de prédire la charge de travail cognitive d'un pilote lors du décollage, afin de mieux comprendre et potentiellement prévenir ces erreurs humaines (Antoine *et al.*, 2022; Ghaderi *et al.*, 2023a). L'étude a mesuré la charge cognitive des pilotes en utilisant plusieurs indicateurs physiologiques, dont la fréquence cardiaque, la dilatation pupillaire et les signaux EEG. Les résultats ont montré qu'une surcharge cognitive peut entraîner des erreurs humaines aux conséquences potentiellement dramatiques, notamment lors des phases critiques du vol comme le

décollage en situation normale ou d'urgence. L'étude a démontré également la faisabilité de la prédiction de la charge mentale des pilotes grâce à des modèles d'apprentissage automatique et à des indicateurs physiologiques.

La deuxième équipe, quant à elle, s'est concentrée sur l'attention des pilotes en vol, un facteur clé pour la sécurité aérienne. Elle a mis en évidence l'impact des facteurs cognitifs et émotionnels sur la performance des pilotes tout en explorant les moyens de surveiller leur attention et leur charge mentale en temps réel. Plus spécifiquement, les chercheurs avaient pour objectif d'analyser l'impact de la charge de travail cognitive sur l'attention des pilotes afin de mieux comprendre les variations de leur concentration en fonction des exigences opérationnelles. Ils ont également examiné si la dilatation pupillaire, mesurée à l'aide d'un eye-tracker, ainsi que l'activité cérébrale enregistrée par un EEG, constituaient des indicateurs fiables de l'engagement et de l'attention des pilotes en situation de vol. Enfin, ils ont identifié les moments critiques au cours desquels l'attention des pilotes diminuait, augmentant ainsi le risque d'erreur. Les résultats ont montré qu'une mauvaise gestion de l'attention pouvait conduire à des oublis de paramètres essentiels ou à une diminution de la conscience situationnelle (Ghaderi *et al.*, 2022a, 2023b).

Afin d'étendre les capacités du pilote synthétique développé dans cette thèse, une première approche serait d'augmenter l'ontologie de référence en y intégrant des données sur la charge mentale et sur l'attention des pilotes.

Puisque le casque EEG et l'équipement mesurant la fréquence cardiaque (heart rate) sont des dispositifs trop intrusifs, une autre approche consisterait à utiliser l'équipement qui mesure la dilatation de la pupille (eye-tracking) pour modéliser et analyser en temps réel la distribution de l'attention du pilote humain. En effet, il existe une corrélation entre la charge mentale et l'attention du pilote. Si une fixation excessive sur un instrument critique ou une perte d'attention est détectée, le pilote synthétique pourrait réorienter l'attention du pilote humain vers des informations prioritaires ou ajuster l'affichage du cockpit pour optimiser la perception situationnelle.

En intégrant ces mécanismes dans son modèle cognitif et en exploitant l'ontologie de référence augmentée, le pilote synthétique pourrait ainsi devenir un assistant adaptatif, capable d'optimiser la charge de travail et d'améliorer la sécurité aérienne.

6.4.4 Validation du système par les experts de l'aviation

Une autre limite importante concerne la validation du pilote synthétique par des acteurs du domaine aéronautique. Bien que le système ait été conçu conformément aux protocoles de l'aviation civile et validé par des méthodes scientifiquement fondées, notamment à l'aide de la Distance d'Édition de Graphe (présentée dans les chapitres précédents), une évaluation du comportement du pilote synthétique par des professionnels tels que des pilotes de ligne, des instructeurs en simulateur et des régulateurs de l'aviation civile demeure nécessaire. Un processus de validation impliquant ces acteurs permettrait d'évaluer la pertinence du modèle en conditions réelles, de mesurer son impact sur la formation des pilotes et d'identifier les ajustements nécessaires pour optimiser son intégration dans les systèmes existants.

En ce qui concerne l'ontologie, elle a été développée durant le projet Pilot-AI. L'équipe projet était constituée, entre autres, d'experts de chez Bombardier et CAE, des entreprises de renom dans le domaine de l'aviation. On y retrouvait des pilotes d'aéronef, des instructeurs en simulateur de vol et bien d'autres experts du domaine. Ces différents acteurs ont participé à des séances de travail collaboratives pour affiner et corriger l'ontologie de référence. Leur retour d'expérience a permis d'ajuster progressivement l'ontologie de référence pour mieux formaliser les actions des pilotes et l'environnement de simulation. De plus, l'ontologie de référence étant également basée sur l'ontologie Air Traffic Management Ontology (atmonto) de la NASA, cette section de l'ontologie a suffisamment été testée et validée par la NASA.

Relativement à la validation du modèle cognitif, elle repose sur les indications faites dans (Raluca, 2013), décrivant comment un modèle cognitif ACT-R est développé en partant d'observations psychologiques générales, en passant par la formation d'hypothèses sur la cognition humaine, puis en intégrant des connaissances spécifiques à un domaine particulier et en le validant (voir Figure 6.10, adaptée de (Raluca, 2013)). La création du pilote synthétique s'est appuyée sur cette démarche.

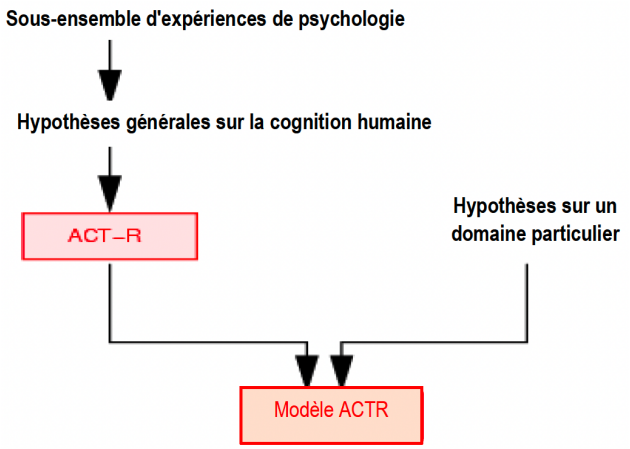


Figure 6.10 – Description de la création d'un modèle ACT-R.

La Figure 6.11, adaptée de (Raluca, 2013), illustre le processus de validation d'un agent cognitif. La validation consiste à comparer les résultats obtenus par le modèle ACT-R à ceux effectués par des humains.

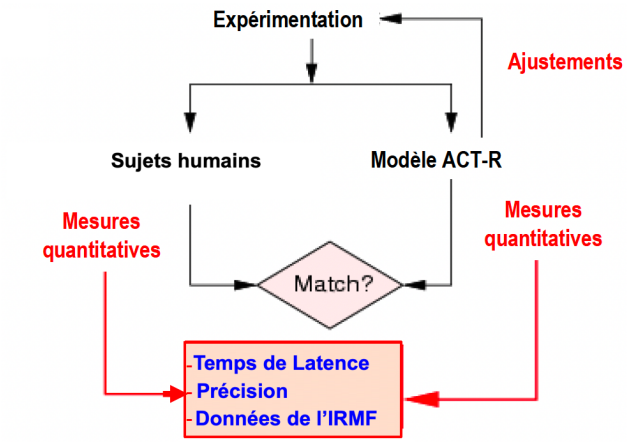


Figure 6.11 – Processus de validation d'un modèle cognitif ACT-R.

La validation actuelle du pilote synthétique a été réalisée selon le processus illustré dans la Figure 6.12. Pour cela, nous avons recueilli les mesures quantitatives produites par l'agent cognitif ACT-R et les avons comparées aux mesures quantitatives expertes, formalisées par l'ontologie de référence, préalablement validée. De plus, comme pour l'ontologie, une validation a été effectuée durant le projet Pilot-AI par des acteurs de l'aviation, constitués d'experts de chez Bombardier et CAE. Ces spécialistes ont participé à des séances de travail collaboratives afin d'affiner le pilote synthétique. Leur retour d'expérience a permis d'ajuster pro-

gressivement ce dernier pour mieux simuler les actions des pilotes, telles que formalisées dans l'ontologie.

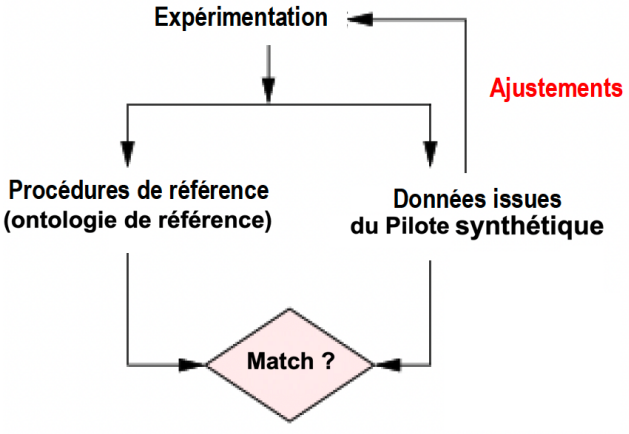


Figure 6.12 – Processus de validation actuel du pilote synthétique à partir de l'ontologie de référence.

À terme, la validation du pilote synthétique suivra le processus illustré dans la Figure 6.13. Elle consistera à recueillir les mesures quantitatives issues de l'agent cognitif ACT-R et à les comparer aux mesures quantitatives obtenues auprès de pilotes humains, relativement à une tâche donnée. Il s'agira notamment d'évaluer si le pilote synthétique exécute correctement sa tâche, tout en tenant compte de contraintes telles que le profil comportemental, l'attention ou la charge cognitive du pilote.

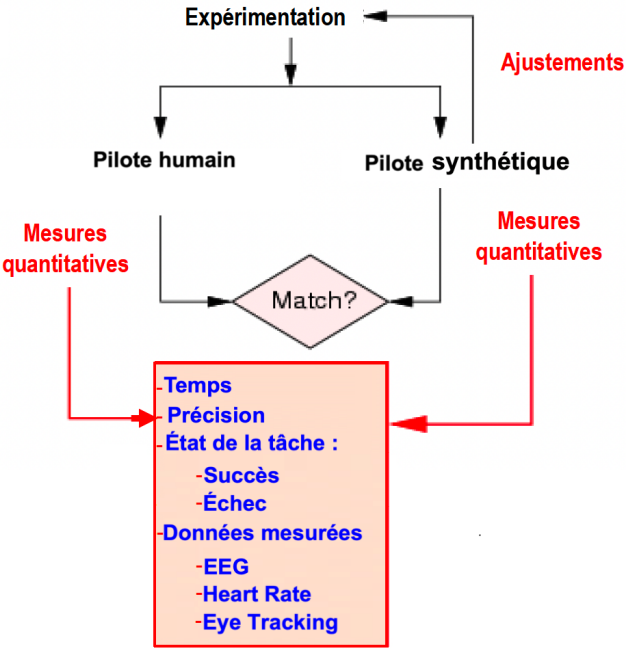


Figure 6.13 – Processus de validation du pilote synthétique intégrant les dimensions cognitive et affective.

6.4.5 Enjeux éthiques et réglementaires

L'intégration d'un assistant cognitif dans la formation et l'aide à la décision des pilotes soulève des enjeux éthiques et réglementaires majeurs, nécessitant une étude approfondie (Hobbs et Li, 2024; Blackett, 2022). L'usage de l'IA dans des environnements critiques comme l'aviation implique des responsabilités partagées entre les concepteurs du système, les compagnies aériennes, les régulateurs et les pilotes eux-mêmes (Tea et Soulé, 2024).

L'une des préoccupations centrales concerne la responsabilité en cas d'erreur (Kumar, 2024). En effet, si le pilote synthétique influence une décision conduisant à un incident ou à un accident, la question de la responsabilité juridique devient complexe : incombe-t-elle au pilote, à l'éditeur du logiciel, à l'organisme de certification ou à l'exploitant aérien ?

Un autre défi majeur réside dans la certification des systèmes autonomes. L'industrie aéronautique est soumise à des standards de sécurité extrêmement stricts, définis par des organismes tels que l'EASA (European Union Aviation Safety Agency) et la FAA (Federal Aviation Administration). La validation d'un assistant cognitif nécessitera des critères rigoureux garantissant sa fiabilité, sa robustesse et son intégration harmonieuse avec les procédures opérationnelles existantes. Des processus d'évaluation continue devront être mis en place pour s'assurer que le pilote synthétique évolue en conformité avec les exigences de sûreté aérienne (Bello *et al.*, 2024).

Par ailleurs, l'acceptabilité des pilotes face à l'automatisation croissante représente un enjeu psychologique et opérationnel fondamental. Si l'IA peut offrir une assistance précieuse dans la gestion de la charge cognitive et la prise de décision en situation de stress, elle ne doit en aucun cas altérer la confiance des pilotes en leurs propres compétences ni engendrer une dépendance excessive aux systèmes automatisés. Des études comportementales et des tests en simulateur seront indispensables pour évaluer l'impact du pilote synthétique sur la performance humaine et adapter son ergonomie aux besoins réels des équipages (Davidoff *et al.*, 2024).

Enfin, une analyse approfondie de ces enjeux, menée en collaboration avec les autorités de régulation aérienne, les acteurs industriels du secteur, les syndicats et les associations de pilotes, serait essentielle pour anticiper les défis et assurer une transition efficace et sécurisée vers l'intégration du pilote synthétique cognitif développé dans cette thèse, pour une aviation toujours plus intelligente et sécurisée.

6.5 Perspectives

Les limites identifiées dans cette recherche ouvrent la voie à plusieurs perspectives d'amélioration et d'extension du pilote synthétique cognitif. Nous présentons ici les directions futures qui pourraient être explorées pour surmonter ces limitations et renforcer la pertinence et l'efficacité du système développé.

6.5.1 Vers une validation en simulateurs de vol certifiés

Les résultats préliminaires obtenus avec X-Plane démontrent la faisabilité technique de la communication entre le pilote synthétique et un environnement de simulation réaliste. La prochaine étape consistera à renforcer cette intégration en permettant au pilote synthétique d'effectuer des actions autonomes basées sur l'inférence de règles de production dérivées de l'analyse des situations perçues. Aussi, l'intégration et l'évaluation du modèle cognitif dans des environnements de simulation plus proches des conditions réelles de vol, comme ceux utilisés par l'industrie aéronautique pour la formation des pilotes, permettraient de mesurer la portabilité du système

6.5.2 Vers une intégration de l'apprentissage automatique pour une adaptation dynamique

Une approche hybride, combinant des règles d'inférence SWRL avec un apprentissage automatique distinct de celui intégré par défaut dans ACT-R, pourrait être envisagée afin d'optimiser à la fois la flexibilité et la précision du modèle cognitif. L'enrichissement du modèle cognitif par des mécanismes d'apprentissage automatique représente une perspective majeure d'évolution. En s'appuyant sur les travaux existants relatifs aux profils comportementaux des pilotes, notamment la distinction entre pilotes à gain élevé et pilotes à gain faible, le pilote synthétique pourrait intégrer ces différentes modalités d'actions et adapter son comportement en conséquence.

6.5.3 Vers une prise en compte de la charge mentale et de l'attention des pilotes

Une perspective essentielle consiste à enrichir le pilote synthétique par l'intégration des connaissances acquises sur la charge mentale et l'attention des pilotes. Concrètement, cela passerait par l'augmentation de l'ontologie de référence pour y inclure des paramètres liés à ces aspects cognitifs et emotionnels. L'exploitation des données d'eye-tracking apparaît comme une approche particulièrement prometteuse, permettant de modéliser en temps réel la distribution de l'attention du pilote. Le pilote synthétique pourrait ainsi de-

venir un assistant cognitif adaptatif, capable d'identifier les situations où l'attention du pilote humain est inadéquate et d'intervenir de manière appropriée. Des mécanismes d'adaptation dynamique pourraient être développés pour que le pilote synthétique ajuste ses interactions en fonction de l'état cognitif du pilote humain, optimisant ainsi la charge de travail et renforçant la sécurité.

6.5.4 Vers une validation approfondie par des experts de l'aviation

Bien que l'ontologie et le pilote synthétique aient bénéficié d'une validation par des experts du domaine dans le cadre du projet Pilot-AI, une validation plus systématique et approfondie reste nécessaire. La mise en place d'un processus de validation continue impliquant une diversité d'acteurs de l'aviation (pilotes, instructeurs, régulateurs, etc.) permettrait d'affiner le système et d'accroître sa pertinence opérationnelle. La transition vers une validation complète, comparant directement les performances du pilote synthétique à celles de pilotes humains dans des tâches spécifiques, constitue une étape cruciale. Cette validation devra notamment évaluer la capacité du système à tenir compte des dimensions cognitives et affectives du pilote. À terme, le pilote synthétique pourrait évoluer vers un véritable coach cognitif, capable non seulement de simuler le comportement des pilotes, mais également de les guider dans leur formation et leur prise de décision.

6.5.5 Vers un traitement des enjeux éthiques et réglementaires

L'implémentation d'un pilote synthétique cognitif dans le domaine aéronautique soulève des défis éthiques et réglementaires majeurs. Il est essentiel d'anticiper les questions de responsabilité juridique en cas d'erreur, d'assurer la transparence des décisions prises par le système et de garantir une conformité aux normes de certification aéronautique établies par des organismes régulateurs comme l'EASA (European Union Aviation Safety Agency) et la FAA (Federal Aviation Administration). Une étude approfondie sur l'impact de l'automatisation cognitive sur la formation et la prise de décision des pilotes devra être menée en concertation avec les organismes de régulation et les parties prenantes de l'industrie aéronautique. Enfin, des études sur l'acceptabilité de ces systèmes par les pilotes et leur influence sur les compétences humaines seront nécessaires afin d'assurer une intégration harmonieuse des technologies d'IA à l'instar du pilote synthétique, dans le pilotage des avions.

6.5.6 Vers un pilote virtuel

Cette thèse s'inscrit dans le cadre du projet Pilot-AI, une collaboration réunissant des institutions académiques (UQAM, UdeM), des partenaires industriels (Bombardier, BMU, CAE) et des organismes de recherche (CRSNG, CRIAQ). L'un des objectifs de ce projet était le développement de pilotes synthétiques capables d'assister les pilotes humains en temps réel.

Deux approches principales ont été explorées, dont :

- Un pilote synthétique basé sur un modèle appris par apprentissage automatique, reproduisant les profils comportementaux de pilotes humains ;
- Un pilote synthétique fondé sur l'architecture ACT-R, s'appuyant sur l'ontologie de référence pour reproduire les actions d'un pilote humain conformément aux standards prescrits par les experts du domaine.

D'autres modèles complémentaires ont été développés pour analyser l'attention du pilote, ses émotions et sa charge cognitive, afin d'évaluer l'impact de ces facteurs sur sa performance. Les résultats du projet ont donné lieu à la publication de plusieurs articles scientifiques, ainsi qu'à la rédaction de mémoires de maîtrise et de thèses de doctorat, dont la présente thèse.

La théorie ACT-R permet à un agent cognitif de percevoir son environnement, de traiter les informations perçues et d'agir en conséquence, tout en intégrant à la fois des dimensions cognitives (attention, charge mentale, apprentissage, résolution de problèmes, etc.) et affectives (émotions, humeurs, sentiments, oubli, etc.). L'intégration harmonieuse de ces différentes modalités au sein du pilote synthétique ACT-R, combinée à une validation par des experts et à la prise en compte des enjeux éthiques et des cadres réglementaires, constitue l'un des objectifs majeurs du projet C-Pilot.

Lancé en 2024 à la suite du projet Pilot-AI, C-Pilot en constitue le prolongement naturel. Il étend la collaboration à d'autres institutions (UQAM, UdeM, UQAC, ULaval, ETS), à de nouveaux partenaires industriels (Bombardier, BMU, Cognitive Group) et à des organismes de recherche (CRSNG, CRIAQ). Son ambition est de concevoir un pilote virtuel, c'est-à-dire un pilote synthétique capable de modéliser l'ensemble des comportements d'un pilote humain, en intégrant ses dimensions cognitive et affective. Plusieurs travaux de recherche, incluant des mémoires de maîtrise et deux thèses de doctorat, sont actuellement en cours afin de tirer parti des avancées du pilote synthétique ACT-R développé dans cette thèse et d'en faire un véritable

pilote virtuel.

6.6 Conclusion

Ce chapitre a présenté une analyse intégrée des résultats de notre recherche, en mettant en lumière les contributions majeures, les limites identifiées, ainsi que des pistes de solutions préliminaires pour y remédier. Il a également ouvert des perspectives pour l'enrichissement des capacités du pilote synthétique cognitif, fondé sur l'architecture ACT-R et l'ontologie de référence.

Menée dans le cadre du projet Pilot-AI, cette recherche visait à améliorer l'assistance cognitive des pilotes. Nous avons décrit le cadre expérimental, analysé les implications informatiques et cognitives de notre approche, et souligné les principaux défis à relever. Enfin, les perspectives proposées ouvrent la voie à des travaux futurs visant à optimiser l'interaction entre l'IA et l'expertise humaine dans le domaine de l'aviation, en vue de concevoir une assistance cognitive à la fois performante et adaptée aux exigences opérationnelles.

CONCLUSION

Cette recherche est partie du constat selon lequel le facteur humain est le maillon faible de la sécurité aérienne. Les statistiques indiquent que 53% des accidents sont causés par des erreurs humaines de pilotage et se produisent principalement pendant les phases de décollage et d'atterrissage. Plus précisément, la phase d'atterrissage, qui représente 4% de la durée du vol, enregistre 25% des accidents mortels, tandis que la phase de décollage, qui représente 2% de la durée du vol, enregistre 30% des accidents mortels (Clifford, 2022).

Pour répondre à cette problématique, nous avons poursuivi tout au long de cette thèse l'objectif de développer un pilote synthétique reposant sur une architecture cognitive et intégrant une ontologie formalisant les connaissances expertes relatives aux procédures de pilotage et à leur environnement d'exécution.

Il convient de souligner que, dans le cadre de cette thèse, notre pilote synthétique s'appuie exclusivement sur la connaissance experte formalisée par l'ontologie de référence ainsi que sur les mécanismes d'apprentissage natifs d'ACT-R pour reproduire fidèlement les comportements d'un pilote humain lors du segment de vol correspondant au décollage.

Notre recherche vise à contribuer à la réduction du nombre d'accidents imputables aux erreurs humaines de pilotage lors de la phase critique du décollage. Pour ce faire, nous avons combiné les avancées en informatique et en sciences cognitives pour créer un système capable de reproduire le comportement d'un pilote humain dans des situations normales et critiques.

Les résultats expérimentaux ont confirmé les hypothèses de recherche formulées au début de cette thèse. D'une part, l'architecture cognitive ACT-R s'est révélée efficace pour modéliser le comportement d'un pilote humain, en intégrant des capacités cognitives telles que la perception, l'attention, la mémoire, la prise de décision et l'action. D'autre part, l'ontologie de référence a démontré sa capacité à structurer les connaissances procédurales complexes liées aux tâches de pilotage d'un avion et à leur environnement d'exécution, en offrant un cadre formel permettant l'exécution automatique des procédures. Enfin, l'intégration de cette ontologie dans l'agent cognitif ACT-R a permis d'améliorer la formation des pilotes, de les assister dans leur prise de décision et de faciliter l'exécution de leurs tâches de pilotage lors du décollage, que ce soit en situation normale ou critique.

Plus spécifiquement, les deux premiers chapitres de ce travail ont permis de structurer notre démarche et de justifier les choix technologiques et méthodologiques retenus pour le développement du pilote synthétique. Le premier chapitre a été consacré à l'analyse des architectures cognitives candidates, en se focalisant sur les modèles les plus reconnus pour modéliser les processus cognitifs humains en situations critiques. Nous y avons examiné cinq architectures majeures : ACT-R, SOAR, CLARION, SPAUN et LEABRA, en mettant en lumière leurs caractéristiques distinctes, leurs forces et leurs limites. Les critères de sélection ayant conduit au choix d'ACT-R comme architecture cognitive retenue y ont également été exposés. Le deuxième chapitre a posé les fondements théoriques du pilote synthétique en détaillant les principes de fonctionnement du pilote automatique et du pilote synthétique, ainsi que leurs différences fondamentales. Il a également justifié le choix du modèle de référence ontologique pour structurer les connaissances nécessaires au pilotage, démontrant comment cette synergie avec ACT-R peut permettre de modéliser fidèlement les processus cognitifs des pilotes humains dans des contextes critiques.

Les chapitres 3 et 4 ont exposé le développement du pilote synthétique, conçu pour simuler les processus cognitifs humains et assister les pilotes dans l'exécution de tâches complexes liées au décollage en situation normale. Le chapitre 3 a montré comment l'ontologie de référence est intégrée dans le pilote ACT-R afin de lui permettre de reproduire le raisonnement et les actions d'un pilote humain lors d'une tâche de pilotage spécifique au décollage. Le chapitre 4 a ensuite élargi ses capacités, en faisant du pilote synthétique un véritable système de coaching cognitif capable de guider les pilotes tout au long de la procédure complète de décollage normal, conformément aux standards établis par les experts du domaine aéronautique.

Une extension significative des capacités du pilote synthétique est introduite au chapitre 5. Après avoir démontré sa compétence à exécuter une procédure de décollage en situation normale, nous avons élargi son champ d'intervention à la gestion des situations anormales susceptibles de survenir lors de cette phase critique du vol, à l'origine d'environ 30% des accidents mortels. Pour ce faire, le système de coaching cognitif précédent a été amélioré, permettant au pilote synthétique d'assister les pilotes humains, non seulement dans leur formation, mais aussi dans leur prise de décision. Cette extension repose sur la consolidation des connaissances procédurales et déclaratives acquises dans les phases précédentes et dote l'agent cognitif de la capacité à détecter les écarts par rapport aux procédures standard, à diagnostiquer les situations et à formuler des recommandations adaptées. Le pilote synthétique est ainsi en mesure de traiter efficacement des scénarios critiques tels qu'une panne moteur après V1, un décollage interrompu, un cisaillement du vent réactif, une alerte TCAS, une double panne moteur avec carburant restant, ou encore une récupération

après un décrochage.

Le chapitre 6 a présenté le pilote synthétique dans un cadre plus vaste, celui du projet Pilot-AI, en mettant en lumière ses limites actuelles ainsi que des pistes de solutions préliminaires à explorer dans le cadre de recherches futures. Plus spécifiquement, il a souligné l'importance de la portabilité du pilote synthétique vers des simulateurs de vol plus réalistes, notamment des simulateurs certifiés, afin d'améliorer la validation du modèle et son applicabilité dans des contextes opérationnels réels. Dans cette perspective, les avancées réalisées pour connecter le pilote synthétique au simulateur de vol X-Plane ont été présentées, de même que les résultats obtenus pour l'interfaçage avec le modèle appris des profils comportementaux des pilotes. Le chapitre a également mis en évidence la nécessité d'enrichir les connaissances de l'agent cognitif en étendant l'ontologie de référence avec des données cognitives et affectives, incluant des facteurs tels que l'attention, la charge mentale et les émotions. Enfin, il a abordé les enjeux éthiques et réglementaires liés à l'adoption de la plateforme par les acteurs du secteur.

Les principales contributions de ce travail s'articulent autour de deux axes majeurs : informatique et cognitif.

Sur le plan informatique, nous avons démontré la faisabilité de l'intégration harmonieuse de l'ontologie de référence dans l'architecture cognitive ACT-R, permettant la modélisation et l'exécution des processus décisionnels d'un pilote. La création et l'utilisation des règles SWRL, combinées au graphe de tâches généré dynamiquement par l'agent cognitif, ont permis d'implémenter un mécanisme de raisonnement adaptatif capable de s'ajuster aux contraintes opérationnelles et aux situations de vol.

Sur le plan cognitif, notre approche a reproduit les processus mentaux d'un pilote humain en modélisant, grâce à l'architecture ACT-R, ses mécanismes de perception, de mémorisation, de traitement et d'action. Grâce à l'intégration d'un modèle de coaching cognitif, l'agent a été capable de détecter et de corriger les erreurs, améliorant ainsi la formation et l'apprentissage des pilotes grâce à une assistance adaptative. Enfin, l'implémentation d'un modèle de mémoire déclarative et procédurale, inspiré des principes d'ACT-R, a renforcé la compréhension du fonctionnement cognitif humain dans un contexte de pilotage.

En termes de perspectives, plusieurs axes d'amélioration et d'approfondissement sont envisagés. Tout d'abord, la validation du système sur des simulateurs de vol certifiés ainsi que sur des plateformes comme X-Plane permettra d'accroître son applicabilité dans des environnements professionnels. Ensuite, l'intégration de techniques d'apprentissage automatique pourrait enrichir le modèle en lui permettant d'évoluer à

partir de données de vol réelles et d'affiner ses recommandations en fonction des profils comportementaux des pilotes. Par ailleurs, l'intégration de paramètres cognitifs et affectifs, tels que l'attention, la charge mentale et les émotions, améliorerait l'adaptabilité du système aux besoins individuels. En outre, une évaluation approfondie du système par des experts de l'aviation civile sera essentielle pour garantir la pertinence et la fiabilité des décisions prises par le pilote synthétique. Enfin, les enjeux éthiques et réglementaires doivent être analysés en détail, notamment en ce qui concerne la responsabilité en cas d'erreur, la supervision humaine, ainsi que la conformité aux normes de certification aéronautique, afin de permettre une intégration sécurisée du pilote synthétique dans des environnements réels.

Pour conclure, la théorie ACT-R permet à un agent cognitif de percevoir son environnement, de traiter les informations perçues et d'agir en conséquence, tout en intégrant les dimensions cognitives (attention, charge mentale, apprentissage, résolution de problèmes, etc.) et affectives (émotions, humeurs, sentiments, oubli, etc.). L'intégration de ces différentes modalités, combinée à une validation par des experts, ainsi qu'à la prise en compte des enjeux éthiques et des cadres réglementaires, permettrait de faire du pilote synthétique un véritable pilote virtuel, c'est-à-dire un agent cognitif capable de modéliser le comportement du pilote humain dans toutes ses dimensions. C'est dans cette perspective que s'inscrit le projet C-Pilot, lancé en 2024 à la suite du projet Pilot-AI, qui réunit plusieurs universités (UQAM, UdeM, UQAC, ULaval, ETS), des partenaires industriels (Bombardier, BMU, Cognitive Group) et des organismes de recherche (CRSNG, CRIAQ) pour mener à bien ce projet ambitieux.

ANNEXE A

LISTE DES ÉQUIPEMENTS DE NAVIGATION UTILISÉS DANS L'EXPÉRIMENTATION

(a) Thrustmaster TCA Sidestick Airbus Edition.

(b) Thrustmaster TCA Quadrant Airbus Edition.

(c) Pédales de gouvernail TFRP.

Figure A.1 – Équipements de navigation utilisés dans le projet Pilot-AI.

ANNEXE B

LISTE DES ÉQUIPEMENTS DE MESURE UTILISÉS DANS L'EXPÉRIMENTATION

(a) Polar H10 (Ceinture pour évaluer la fréquence cardiaque).

(b) Gazepoint GP3 (équipement pour contrôler la dilatation de la pupille).

(c) Casque EEG NC (casque pour mesurer l'activité cérébrale).

Figure B.1 – Équipements de mesure de l'activité cérébrale, cardiaque et oculaire, utilisés dans le projet Pilot-AI.

ANNEXE C

EXPÉRIMENTATION AVEC UN PILOTE, UN COPILOTE ET DES PARTICIPANTS

Figure C.1 – Expérimentation : Pilote, copilote et participant.

ANNEXE D

ENVIRONNEMENT DU POSTE DE PILOTAGE

Figure D.1 – Environnement du poste de pilotage.

ANNEXE E

UN EXTRAIT DU DICTIONNAIRE DE VARIABLES UTILISÉ PAR LE DICTIONNAIRE DE L'API DU SIMULATEUR X-PLANE

```
:
tag: "sim/time/zulu_time_sec"
frequency: 10
description: "EMPTY"
aircraftDescription:
tag: "sim/aircraft/view/acf_descrip[0][40]"
frequency: 10
description: "EMPTY"
StickPositionY:
tag: "a320/Aircraft/Cockpit/Panel/SidestickLonL"
frequency: 10
description: "EMPTY"

StickPositionX:
tag: "a320/Aircraft/Cockpit/Panel/SidestickLatL"
frequency: 10
description: "EMPTY"

engineFireSwitch:
tag: "a320/Overhead/FireEngine1"
frequency: 1
description: ""
engine2FireSwitch:
tag: "a320/Overhead/FireEngine2"
frequency: 1
description: ""
engine2MasterSwitchPosition:
```

```
tag: "a320/Pedestal/EngineMaster2_switch+"
frequency: 1
description: ""
engine1MasterSwitchPosition:
tag: "a320/Pedestal/EngineMaster1_switch+"
frequency: 1
description: ""
egtInCelsius:
tag: "sim/aircraft/engine/acf_EGT_is_C"
frequency: 1
description: ""
AutoThrustMode:
tag: "a320/Panel/FCU_AutoThrust"
frequency: 1
description: ""
AutoPilotMasterSwitch:
tag: "a320/Panel/FCU_AutoPilot1"
frequency: 1
description: ""
agent2Switch:
tag: "a320/Overhead/FireEngine1_Agent2"
frequency: 1
description: ""
agent1Switch:
tag: "a320/Overhead/FireEngine1_Agent1"
frequency: 1
description: ""
ParkBrakePosition:
tag: "sim/flightmodel/controls/parkbrake"
frequency: 1
description: ""
AutoBrakeLeverPosition:
```

```
tag: "a320/Panel/BrakeAuto1"
frequency: 1
description: ""
BrakeLeft:
tag: "sim/cockpit2/controls/left_brake_ratio"
frequency: 1
description: ""
BrakeRight:
tag: "sim/cockpit2/controls/left_brake_ratio"
frequency: 1
description: ""
LandingGearLeverPosition:
tag: "sim/cockpit2/controls/gear_handle_down"
frequency: 1
description: ""
AirSpeed:
tag: "sim/flightmodel/position/indicated_airspeed"
frequency: 10
description: ""
VerticalIndication:
tag: "sim/cockpit/autopilot/vertical_velocity"
frequency: 10
description: ""
RadioAltitude:
tag: "sim/cockpit2/gauges/indicators/radio_altimeter_height_ft_pilot"
frequency: 10
description: ""
TcasMode:
tag: "sim/cockpit2/EFIS/EFIS_tcas_on"
frequency: 10
description: ""
RudderDirection:
```

```
tag: "sim/flightmodel2/controls/rudder_trim"
frequency: 10
description: "Rudder trim values"
MasterSwitchWarningSwitchPress:
tag: "sim/cockpit/warnings/annunciator_test_pressed"
frequency: 10
description: ""
RadioTransmitting:
tag: "sim/atc/user_aircraft_transmitting"
frequency: 10
description: ""
SurfaceWind:
tag: "sim/weather/wind_speed_kt[0]"
frequency: 10
description: ""
CrossWind:
tag: "sim/weather/wind_speed_kt[1]"
frequency: 10
description: ""
Engine1Failure:
tag: "sim/operation/failures/rel_engfai1"
frequency: 10
description: ""
Engine2Failure:
tag: "sim/operation/failures/rel_engfai2"
frequency: 10
description: ""
LandingGearPosition:
tag: "sim/aircraft/prop/acf_prop_gear_rat"
frequency: 10
description: ""
AirSpeed:
```

```
  tag: "sim/flightmodel/position/indicated_airspeed"
  frequency: 10
  description: ""
LeftThrustLever_write:
  tag: "sim/cockpit2/engine/actuators/throttle_ratio[0]"
  frequency: 10
  description: "Engine 1 Thrust value ratio between 0-1"
RightThrustLever_write:
  tag: "sim/cockpit2/engine/actuators/throttle_ratio[1]"
  frequency: 10
  description: "Engine 2 Thrust value ratio between 0-1"
LeftThrustLever_Read:
  tag: "a320/Aircraft/Cockpit/Pedestal/EngineLever1"
  frequency: 10
  description: "Engine 2 Thrust value ratio between 0-1"
RightThrustLever_Read:
  tag: "a320/Aircraft/Cockpit/Pedestal/EngineLever2"
  frequency: 10
  description: "Engine 2 Thrust value ratio between 0-1"
AircraftDirection:
  tag: "sim/cockpit/gyros/psi_ind_degm3"
  frequency: 10
  description: ""
Altitude:
  tag: "sim/flightmodel/misc/h_ind"
  frequency: 10
  description: ""
FlapLever:
  tag: "sim/multiplayer/controls/flap_request"
  frequency: 10
  description: "FLAP values"
TcasAlert:
```

```
156 tag: "sim/cockpit2/tcas/indicators/tcas_alert"
157 frequency: 10
158 description: "TCAS Alert"
159 Pitch:
160 tag: "sim/cockpit/gyros/the_vac_ind_deg"
161 frequency: 10
162 description: ""
163 WindShearWarning:
164 tag: "sim/cockpit2/annunciators/windshear_warning"
165 frequency: 10
166 description: "Windshear warning values"
167
```

BIBLIOGRAPHIE

Adomavicius, G. et Tuzhilin, A. (2015). Context-aware recommender systems. 191–226. http://dx.doi.org/10.1007/978-1-4899-7637-6_6

Airbus (2006a). *Flight Crew Operating Manual : Part 3 Flight Operations*. A319 / A320 / A321.

Airbus (2006b). *Flight Crew Operating Manual : Part 4 FMGS Pilot's Guide*. A319 / A320 / A321.

Airbus (2012). *Quick Reference Handbook*. A318 / A319 / A320 / A321.

Anderson, J. R. (2007). *How Can the Human Mind Occur in the Physical Universe ?* Oxford : Oxford University Press. http://dx.doi.org/10.1093/acprof:oso/9780195324259.001.0001

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. et Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*(4), 1036–1060. http://dx.doi.org/10.1037/0033-295X.111.4.1036

Anderson, J. R., Fincham, J. M. et Douglass, S. (1997). The role of examples and rules in the acquisition of a cognitive skill. *Journal of Experimental Psychology : Learning, Memory, and Cognition*, *23*(4), 932–945. http://dx.doi.org/10.1037/0278-7393.23.4.932

Anderson, J. R. et Lebiere, C. (2003). The newell test for a theory of cognition. *Behavioral and Brain Sciences*, *26*(5), 587–601.

Antoine, M., Abdessalem, H. et Frasson, C. (2022). Cognitive workload assessment of aircraft pilots. *Journal of Behavioral and Brain Science*, *12*, 474–484. http://dx.doi.org/10.4236/jbbs.2022.1210027

Aprendamos Aviación (2022). Aviation : Commandes de vol. Laeronef-5. Accessed 2025-02-26. Récupéré de https://laeronef-5.aprendamos-aviacion.com/2022/11/aeronef-gouvernes-de-vol-flight-control.html

Armstrong, S. A. et Herr, M. J. (2023). *Physiology, Nociception* (updated 2023 may 1 éd.). Treasure Island (FL) : StatPearls Publishing. Disponible à : https://www.ncbi.nlm.nih.gov/books/NBK551562/.

Asselman, A., Aammou, S. et Nasseh, A. E. (2015). Comparative study of cognitive architectures. *International Research Journal of Computer Science*, *2*(9), 8–13.

Atkinson, R. C. et Shifrin, R. M. (1968). *Human Memory : A Proposed System and Its Control Processes*. Stanford, California : Stanford University.

Baars, B. J. (1988). *A Cognitive Theory of Consciousness*. Cambridge, MA : Cambridge University Press.

Baker, M. (2020). President's position : giving george a break. Aircraft Owners and Pilots Association. Accessed 2025-02-26. Récupéré de https://www.aopa.org/news-and-media/all-news/2020/presidents-position-giving-george-a-break

Baomar, H. et Bentley, P. J. (2016). An intelligent autopilot system that learns piloting skills from human pilots by imitation. Dans *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, 1023–1031. IEEE.

Barr, A. et Feigenbaum, E. A. (2014). *The Handbook of Artificial Intelligence : Volume 1* (réimprimée éd.). Butterworth-Heinemann.

Bastien (2017). Cognitivisme : notre intelligence est-elle artificielle ? *Artificiel.net*. Récupéré le 13 décembre 2019 de http://www.artificiel.net/cognitivisme.

Bellman, R. E. (1978). *An Introduction to Artificial Intelligence : Can Computers Think ?* San Francisco : Boyd & Fraser.

Bello, H., Geißler, D., Ray, L. S. S., Müller-Divéky, S., Müller, P., Kittrell, S., Liu, M., Zhou, B. et Lukowicz, P. (2024). Towards certifiable ai in aviation : Landscape, challenges, and opportunities. *arXiv preprint*. Preprint available on arXiv, http://dx.doi.org/10.48550/arXiv.2409.08666. Récupéré de https://doi.org/10.48550/arXiv.2409.08666

Billings, C. (1996). *Aviation Automation : The Search for A Human-centered Approach* (1st éd.). CRC Press. http://dx.doi.org/10.1201/9781315137995

Blackett, C. (2022). The ethics of ai in autonomous transport. Dans *Proceedings of the 32nd European Safety and Reliability Conference (ESREL)*. Risk Pilot AB. Conference : 32nd European Safety and Reliability Conference, http://dx.doi.org/10.3850/978-981-18-5183-4_J03-05-453-cd. Récupéré de https://doi.org/10.3850/978-981-18-5183-4_J03-05-453-cd

Blouw, P., Eliasmith, C. et Tripp, B. (2016a). A scaleable spiking neural model of action planning. Dans A. Papafragou, D. Grodner, D. Mirman, et J. Trueswell (dir.). *Proceedings of the 38th Annual Conference of the Cognitive Science Society*, 1583–1588., Philadelphia, Pennsylvania. Cognitive Science Society. Récupéré de https://mindmodeling.org/cogsci2016/papers/0279/index.html

Blouw, P., Solodkin, E., Thagard, P. et Eliasmith, C. (2016b). Concepts as semantic pointers : A framework and computational model. *Cognitive Science*, *40*(5), 1128–1162. http://dx.doi.org/10.1111/cogs.12265

Bodeen, C. (2011). Commercial applications of X-Plane - Part Two.

Boole, G. (1854). *An Investigation of the Laws of Thought, on Which Are Founded the Mathematical Theories of Logic and Probabilities*. London : Walton and Maberly. http://dx.doi.org/10.5962/bhl.title.29413

Bouzekri, E., Canny, A., Fayollas, C., Martinie, C., Palanque, P., Barboni, E., Deleris, Y. et Gris, C. (2019). Engineering issues related to the development of a recommender system in a critical context : Application to interactive cockpits. *International Journal of Human-Computer Studies*, *121*, 122–141.

Boy, G. A. (2017). Human-centered design of complex systems : An experience-based approach. *Design Science*, *3*, e8. http://dx.doi.org/10.1017/dsj.2017.8

Brasoveanu, A. et Dotlačil, J. (2020). *Computational Cognitive Modeling and Linguistic Theory*. Springer Nature.

Brick, T. et Scheutz, M. (2007). Incremental natural language processing for hri. Dans *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI '07)*, 263–270., New York, NY, USA. Association for Computing Machinery. http://dx.doi.org/10.1145/1228716.1228752

Byrne, M. D., Kirlik, A., Fleetwood, M. D., Huss, D. G., Kosorukoff, A., Lin, R.-S. et Fick, C. S. (2004). A closed-loop, act-r approach to modeling approach and landing with and without synthetic vision system (svs) technology. Dans *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 48, 2111–2115. http://dx.doi.org/10.1177/154193120404801707

Cacciabue, P. C. (1998). *Modelling and Simulation of Human Behaviour in System Control*. Advances in Industrial Control. Springer.

Charniak, E. et McDermott, D. V. (1985). *Introduction to Artificial Intelligence*. Boston : Addison-Wesley Longman Publishing Co.

Chong, H.-Q., Tan, A.-h. et Ng, G.-W. (2009). Integrated cognitive architectures : A survey. *Artificial Intelligence Review*, *28*(2), 103–130. Research Collection School Of Information Systems.

Choo, X. et Eliasmith, C. (2013). General instruction following in a large-scale biologically plausible brain model. Dans *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 35. Récupéré de https://escholarship.org/uc/item/6p45n14b

Clifford, L. (2022). The most common causes of aviation accidents. *National Law Review*, *X*(343).

Corker, K. M. et Smith, B. R. (1993). An architecture and model for cognitive engineering simulation analysis : Application to advanced aviation automation. Dans *9th Computing in Aerospace Conference*. American Institute of Aeronautics and Astronautics. http://dx.doi.org/10.2514/6.1993-4660

Courtemanche, M.-A., Tato, A. et Nkambou, R. (2022). Ontological reference model for piloting procedures. Dans S. Crossley et E. Popescu (dir.). *Intelligent Tutoring Systems*, volume 13284 de *Lecture Notes in Computer Science*, 1–12. Springer, Cham. http://dx.doi.org/10.1007/978-3-031-09680-8_9

Courtemanche, M.-A., Tato, A. et Nkambou, R. (2023). Automatic execution of the ontological piloting procedures. Dans C. Frasson, P. Mylonas, et C. Troussas (dir.). *Augmented Intelligence and Intelligent Tutoring Systems*, volume 13891 de *Lecture Notes in Computer Science*, 29–41., Cham. Springer. http://dx.doi.org/10.1007/978-3-031-32883-1_3

Cox, M. T., Alavi, Z., Dannenhauer, D., Eyorokon, V., Munoz-Avila, H. et Perlis, D. (2016). Midca : A metacognitive, integrated dual-cycle architecture for self-regulated autonomy. Dans *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*. http://dx.doi.org/10.1609/aaai.v30i1.9886

Davidoff, A., Vonderhaar, L., Caldwell, A., Procko, T. et Ochoa, O. (2024). Role, needs, and state of cognitive assistants in single-pilot operations. *Journal of Aerospace Information Systems*, *21*(12), [pages not specified]. Published Online : 7 Oct 2024, http://dx.doi.org/10.2514/1.I011432. Récupéré de https://doi.org/10.2514/1.I011432

Dehais, F., Roy, R. N. et Scannella, S. (2019). Inattentional deafness to auditory alarms : Inter-individual differences, electrophysiological signature and single trial classification. *Behavioural Brain Research*, *360*, 51–59. http://dx.doi.org/10.1016/j.bbr.2018.11.045

Dekker, S. (2014). *The Field Guide to Understanding 'Human Error'* (3rd éd.). CRC Press. http://dx.doi.org/10.1201/9781317031833

Dilmegani, C. et Palazoğlu, M. (2025). Most common soar use cases : 7 real-life examples in 2025. *AIMultiple Research*.

Dismukes, R. K., Berman, B. A. et Loukopoulos, L. D. (2007). *The Limits of Expertise : Re-thinking Pilot Error and the Causes of Airline Accidents*. Aldershot, Hampshire, England : Ashgate.

Droit-Volet, S. (2025). Perception du temps. Disponible à : https://www.universalis.fr/encyclopedie/perception-du-temps/.

Dubois, D. (2007). *Constructing an agent equipped with an artificial consciousness : Application to an intelligent tutoring system*. (Thèse de doctorat). Université du Québec à Montréal, Montréal, Québec, Canada.

Duperrin, B. (2021). Comment fonctionne le pilote automatique ? TravelGuys. Accessed 2025-02-26. Récupéré de https://www.travelguys.fr/2021/07/27/comment-fonctionne-le-pilote-automatique/

Eliasmith, C. (2012). A large-scale model of the functioning brain. *Science*, *338*, 1202–1205. http://dx.doi.org/10.1126/science.1225266

Eliasmith, C. (2013). *How to Build a Brain : A Neural Architecture for Biological Cognition*. Oxford : Oxford University Press.

Estes, S., Burns, K., Helleberg, J., Long, K., Stein, J. et Pollack, M. (2016). Digital copilot : Cognitive assistance for pilots. Dans *Proceedings of the AAAI Fall Symposium on Cognitive Assistance in Government and Public Sector Applications*.

Evans, J. S. B. T. et Stanovich, K. E. (2013). Dual-process theories of higher cognition : Advancing the debate. *Perspectives on Psychological Science*, *8*(3), 223–241.

Faghihi, U. (2011). *The use of emotions in the implementation of various types of learning in a cognitive agent*. (Thèse de doctorat). Université du Québec à Montréal, Montréal, Québec, Canada.

Faghihi, U., Fournier-Viger, P. et Nkambou, R. (2013). Celts : A cognitive tutoring agent with human-like learning capabilities and emotions. In A. Peña-Ayala (dir.), *Intelligent and Adaptive Educational-Learning Systems*, volume 17 de *Smart Innovation, Systems and Technologies*. Berlin, Heidelberg : Springer

FasterCapital (2024). Cognitive modeling : Exploring mental processes through abm techniques. Récupéré le 2025-02-27 de https://fastercapital.com/content/Cognitive-modeling--Exploring-Mental-Processes-through-ABM-Techniques.html

Federal Aviation Administration (2014). *Automated Flight Controls in Advanced Avionics Handbook*. U.S. Department of Transportation, Federal Aviation Administration. Retrieved 20 February 2014, chapter 4, 68-83

Flavell, J. H. (1979). Metacognition and cognitive monitoring : a new area of cognitive-developmental inquiry. *American Psychologist*, *34*(10), 906–911. http://dx.doi.org/10.1037/0003-066X.34.10.906

Fodor, J. A. (1975). *The Language of Thought*. Cambridge, MA : Harvard University Press.

Foyle, D. C. et Hooey, B. L. (dir.) (2007). *Human Performance Modeling in Aviation* (1 éd.). CRC Press. http://dx.doi.org/10.1201/9781420062984

Franklin, S. (2003). Ida : A conscious artifact ? *Journal of Consciousness Studies*, *10*(4-5), 47–66.

Gagnon-Roy, M., Bier, N., Le Dorze, G., Boulé-Riley, S., Paquette, G., Couture, M. et Bottari, C. (2024). Cognitive assistance to support individuals with traumatic brain injury using a minimal and personalised approach : A conversion mixed methods study using video analysis. *Australian Occupational Therapy Journal*, *71*(1), 35–51. PMID : 37799014, http://dx.doi.org/10.1111/1440-1630.12906

Gao, X., Xiao, B., Tao, D. et Li, X. (2010). A survey of graph edit distance. *Pattern Analysis and Applications*, *13*(1), 113–129. http://dx.doi.org/10.1007/s10044-008-0141-y

Garnelo, M. et Shanahan, M. (2019). Reconciling deep learning with symbolic artificial intelligence : representing objects and relations. *Current Opinion in Behavioral Sciences*, *29*, 17–23. Artificial Intelligence, http://dx.doi.org/https://doi.org/10.1016/j.cobeha.2018.12.010. Récupéré de https://www.sciencedirect.com/science/article/pii/S2352154618301943

Ghaderi, M., Abdessalem, H. B., Antoine, M. et Frasson, C. (2023a). Estimation of piloting attention level based on the correlation of pupil dilation and eeg. Dans *International Conference on Intelligent Tutoring Systems*, 381–390.

Ghaderi, M., Abdessalem, H. B. et Frasson, C. (2022a). An analysis of mental workload involved in piloting tasks. Dans *Novel & Intelligent Digital Systems : Proceedings of the 2nd International Conference (NiDS 2022)*, 211–220.

Ghaderi, M., Courtemanche, M.-A., Ben Abdessalem, H., Nkambou, R. et Frasson, C. (2022b). Attentional tasks model : A focus group approach. Dans *Neural Information Processing*, volume 13284 de *Lecture Notes in Computer Science*, 297–307., Cham. Springer. http://dx.doi.org/10.1007/978-3-031-17601-2_29

Ghaderi, M., Khalaj, A. B., Abdessalem, H. B. et Frasson, C. (2023b). Attention assessment of aircraft pilots using eye tracking. Dans *International Conference on Intelligent Tutoring Systems*, 209–219.

Gluck, K. A., Ball, J. T., Krusmark, M. A., Rodgers, S. M. et Purtee, M. D. (2003). A computational process model of basic aircraft maneuvering. Dans F. Detje, D. Doerner, et H. Schaub (dir.). *Proceedings of the Fifth International Conference on Cognitive Modeling*, 117–122., Bamberg, Germany.

González-Santamarta, M. A., Rodrıguez-Lera, F. J., Matellan-Olivera, V. *et al.* (2025). Sailor : perceptual anchoring for robotic cognitive architectures. *Scientific Reports*, *15*, 113. http://dx.doi.org/10.1038/s41598-024-84071-2

Gray, W. D. et Fu, W.-T. (2004). Soft constraints in interactive behavior : The case of ignoring perfect knowledge in-the-world for imperfect knowledge in-the-head. *Cognitive Science*, *28*(3), 359–382. http://dx.doi.org/10.1016/j.cogsci.2003.12.001

Grillner, S. et Robertson, B. (2015). The basal ganglia downstream control of brainstem motor centres–an evolutionarily conserved strategy. *Current Opinion in Neurobiology*, *33*, 47–52. http://dx.doi.org/10.1016/j.conb.2015.01.019

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, *5*(2), 199–220.

Haroon, K. (2020). A320 abnormal procedures. https : //www.theairlinepilots.com/forumarchive/a320/a320-abnormal-procedures.pdf.

Harris, D. (2011). *Human Performance on the Flight Deck* (1st éd.). CRC Press. http://dx.doi.org/10.1201/9781315252988

Haugeland, J. (1985). *Artificial Intelligence : The Very Idea*. Cambridge : MIT Press.

Hebb, D. O. (1949). *The Organization of Behavior : A Neuropsychological Theory*. Wiley.

Heudin, J.-C. (2017). *Intelligence Artificielle : manuel de survie*. Science eBook.

Hobbes, T. (1651). *Leviathan*. London : Andrew Crooke.

Hobbs, K. L. et Li, B. (2024). Safety, trust, and ethics considerations for human-ai teaming in aerospace control. Dans *AIAA SciTech Forum*, AIAA 2024-2583. Published Online : 4 Jan 2024, http://dx.doi.org/10.2514/6.2024-2583. Récupéré de https://doi.org/10.2514/6.2024-2583

Hudlicka, E. (2004). Beyond cognition : Modeling emotion in cognitive architectures. Dans *International Conference on Cognitive Modelling*. Récupéré de https://api.semanticscholar.org/CorpusID:15966621

Hélie, S. (2007). *Modélisation de l'apprentissage ascendant des connaissances explicites dans une architecture cognitive hybride*. (Thèse de doctorat). Université du Québec à Montréal, Montréal, Québec, Canada.

Héroux, M. E., Butler, A. A., Robertson, L. S., Fisher, G. et Gandevia, S. C. (2022). Proprioception : a new look at an old concept. *Journal of Applied Physiology*. Dimensions ID : pub.1145430981, http://dx.doi.org/10.1152/japplphysiol.00809.2021

Insaurralde, C. et Blasch, E. (2019). Uncertainty in avionics analytics ontology for decision-making support. *Journal of Advances in Information Fusion*, *13*(2), 255–274.

Insaurralde, C. C. et Blasch, E. (2016). Ontological knowledge representation for avionics decision-making support. Dans *35th Digital Avionics Systems Conference (DASC)*, Sacramento, CA, USA.

Insaurralde, C. C. et Blasch, E. (2018). Uncertainty in avionics analytics ontology for decision-making support. *Journal of Advances in Information Fusion (JAIF)*, *13*(2), 255–274.

Insaurralde, C. C. et Blasch, E. (2022). Situation awareness decision support system for air traffic management using ontological reasoning. *AIAA Journals*. http://dx.doi.org/10.2514/1.I010989

International Air Transport Association (2024). *Annual Review 2024*. Rapport technique, International Air Transport Association, Dubai, United Arab Emirates. 80th Annual General Meeting and World Air Transport Summit.

IxDF, I. (2020). What is cognitive modeling ? Récupéré le 2025-02-27 de https://www.interaction-design.org/literature/topics/cognitive-modeling

Jones, R. M., Lebiere, C. et Crossman, J. A. (2007). Comparing modeling idioms in act-r and soar. Dans *Proceedings of the Eighth International Conference on Cognitive Modeling*, Ann Arbor, MI.

Kahneman, D. (2011). *Thinking, Fast and Slow*. New York : Farrar, Straus and Giroux.

Kajić, I., Gosmann, J., Stewart, T. C., Wennekers, T. et Eliasmith, C. (2017). A spiking neuron model of word associations for the remote associates test. *Frontiers in Psychology*, *8*, 99. http://dx.doi.org/10.3389/fpsyg.2017.00099

Keller, R. M. (2017a). *Air Traffic Management (ATM) Ontology : Includes Core Classes and Properties*. Rapport technique, National Aeronautics and Space Administration. Retrieved from https://data.nasa.gov/ontologies/atmontoCore/doc/.

Keller, R. M. (2017b). *The NASA Air Traffic Management Ontology : Technical Documentation*. Technical Memo NASA/TM-2017-219526, National Aeronautics and Space Administration.

Keller, R. M. (2018). *The NASA Air Traffic Management Ontology (atmonto) (Version 1.0)*. Rapport technique, National Aeronautics and Space Administration. Retrieved from https://data.nasa.gov/ontologies/atmonto.

Kelley, T. (2003). Symbolic and sub-symbolic representations in computational models of human cognition : what can be learned from biology ? *Theory & Psychology*, *13*(6), 847–860.

Kieras, D. E. et Meyer, D. E. (1997). An overview of the epic architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, *12*(4), 391–438. http://dx.doi.org/10.1207/s15327051hci1204_4

Kotseruba, I. et Tsotsos, J. (2020). 40 years of cognitive architectures : core cognitive abilities and practical applications. *Artificial Intelligence Review*, 17–94.

Kozak, S. (2019). Pilot training innovation to assess and monitor cognitive load capacity. Modern Military Training.

Kumar, A. (2024). Exploring ethical considerations in ai-driven autonomous vehicles : Balancing safety and privacy. *Journal of Artificial Intelligence and Global Solutions*, *2*(1), 138. http://dx.doi.org/10.60087/jaigs.v2i1.p138. Récupéré de https://doi.org/10.60087/jaigs.v2i1.p138

Kurzweil, R. (1990). *The Age of Intelligent Machines*, volume 579. Cambridge : MIT Press.

Laird, J. E. (2012). *The Soar Cognitive Architecture*. MIT Press.

Laminar Research (2017). *X-Plane 11 desktop manual*. Laminar Research, Columbia, SC

Langley, P., Laird, J. E. et Rogers, S. (2009). Cognitive architectures : Research issues and challenges. *Cognitive Systems Research*, *10*(2), 141–160.

Larue, O. (2015). *Une architecture cognitive inspirée des théories des processus duaux pour une interaction fluide des comportements réactifs et délibératifs*. (Thèse de doctorat). Université du Québec à Montréal, Montréal, Québec, Canada.

Lieto, A., Bhatt, M., Oltramari, A. et Vernon, D. (2018). The role of cognitive architectures in general artificial intelligence. *Cognitive Systems Research*, *48*, 1–3.

Mahmood, T. et Ricci, F. (2009). Improving recommender systems with adaptive conversational strategies. Dans C. Cattuto, G. Ruffo, et F. Menczer (dir.). *Hypertext*, 73–82.

McCulloch, W. S. et Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, *5*, 115–133.

McGill (2016). Le cerveau à tous les niveaux. *Sciences cognitives*, p. 61. Récupéré le 02 avril 2025 de https://thebrain.mcgill.ca/flash/capsules/pdf_articles/reseau_neurones.pdf.

McSherry, F. et Mironov, I. (2009). Differentially private recommender systems : Building privacy into the net. Dans *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 627–636. ACM.

Miller, G. A. (1956). The magical number seven, plus or minus two : some limits on our capacity for processing information. *Psychological Review*, *63*(2), 81–97.

Newell, A. (1980). Physical symbol systems. *Cognitive Science*, *4*(2), 135–183.

Newell, A. (1990). *Unified Theories of Cognition*. Harvard University Press.

Newell, A. (1992). Précis of unified theories of cognition. *Behavioral and Brain Sciences*, *15*, 425–492.

Nicolas, S. (2002). *La mémoire*. Topos. Paris : Dunod.

Nilsson, N. J. (1998). *Artificial Intelligence : A New Synthesis*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc.

OACI (2016). *Plan pour la sécurité de l'aviation dans le monde, 2017-2019*. Rapport technique, OACI

Oliver, W. K., Christoph, V., Laurens, R. K., Marc, H., Z., T. O. et R., N. (2020). Tracing pilots' situation assessment by neuroadaptive cognitive modeling. *Frontiers in Neuroscience*, *14*, 795.

Oliver, W. K., Halbrügge, M. et Russwinkel, N. (2019). Act-r model for cognitive assistance in handling flight deck alerts. Dans *International Conference on Cognitive Modelling*, Montreal, Canada.

Oltramari, A. et Lebiere, C. (2011). Mechanisms meet content : Integrating cognitive architectures and ontologies. Dans *AAAI Fall Symposium : Advances in Cognitive Systems*. Récupéré de https://api.semanticscholar.org/CorpusID:12466118

Oltramari, A. et Sun, R. (2025). Dual-process theories, cognitive architectures, and hybrid neural-symbolic models. *Neurosymbolic Artificial Intelligence*, *1*. http://dx.doi.org/10.3233/NAI-240720

O'Reilly, R. C. (1996). *The LEABRA Model of Neural Interactions and Learning in the Neocortex*. (Thèse de doctorat). Carnegie Mellon University, Department of Psychology, Pittsburgh, PA. Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

O'Reilly, R. C., Hazy, T. E. et Herd, S. A. (2017). The leabra cognitive architecture : How to play 20 principles with nature and win ! In S. E. F. Chipman (dir.), *The Oxford Handbook of Cognitive Science* 91–115. Oxford University Press.

Osta-Vélez, M. (2014). Logique, raisonnement et rationalité. http://dx.doi.org/10.13140/RG.2.1.2863.1449

Otelli, J. (2022). *Erreurs de pilotage*, volume 16 de *Histoires Authentiques*. Jpo Altipresse.

Ozturk, P. (2009). Levels and types of action selection : the action selection soup. *Adaptive Behavior*, *17*, 537–554.

Pietracupa, M., Ben Abdessalem, H. et Frasson, C. (2023). An approach to automatic flight deviation detection. Dans *Augmented Intelligence and Intelligent Tutoring Systems*, volume 13891 de *Lecture Notes in Computer Science*. http://dx.doi.org/10.1007/978-3-031-32883-1_47

Poole, D. I., Goebel, R. G. et Mackworth, A. K. (1998). *Computational Intelligence*. New York : Oxford University Press.

Raluca, B. (2013). About act-r. http://act-r.psy.cmu.edu/wordpress/wp-content/uploads/2012/09/buffers.gif. Consulté le 22 février 2025.

Rasmussen, D., Voelker, A. et Eliasmith, C. (2017). A neural model of hierarchical reinforcement learning. *PLoS One*, *12*(7), e0180234. http://dx.doi.org/10.1371/journal.pone.0180234

Rehling, J., Lovett, M., Lebiere, C., Reder, L. M. et Demiral, B. (2004). Modeling complex tasks : An individual difference approach. Dans *Proceedings of the 26th Annual Conference of the Cognitive Science Society*, 1137–1142., Chicago, USA.

Rich, E. et Knight, K. (1991). *Artificial Intelligence*. New York : McGraw-Hill.

Ritter, F. E., Tehranchi, F. et Oury, J. D. (2018). Act-r : A cognitive architecture for modeling cognition. *Wiley Interdisciplinary Reviews : Cognitive Science*, *10*(3), e1488. http://dx.doi.org/10.1002/wcs.1488

Rosenblatt, F. (1958). The perceptron : A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408.

RTCA. (2012). *DO-178C, Software Considerations in Airborne Systems and Equipment Certification*. Washington, DC : RTCA, Inc.

Rubinoff, R. et Lehman, J. (1994). Real-time natural language generation in nl-soar. Dans *Proceedings of the Seventh International Workshop on Natural Language Generation*, 199–206. http://dx.doi.org/10.3115/1641417.1641440

Russell, S. J. et Norvig, P. (2020). *Artificial Intelligence : A Modern Approach* (4th éd.). Boston : Pearson.

Salas, E., Maurino, D. et Curtis, M. (2010). Human factors in aviation : An overview. In *Human Factors in Aviation* 3–19. San Diego, CA : Academic Press, (2 éd.).

Salvucci, D. D. (2001). An integrated model of eye movements and visual encoding. *Cognitive Systems Research*, *1*(4), 201–220.

Salvucci, D. D. (2006). Modeling driver behavior in a cognitive architecture. *Human Factors*, *48*(2), 362–380. http://dx.doi.org/10.1518/001872006777724417

Sarter, N. et Woods, D. (1995). How in the world did we ever get into that mode ? mode error and awareness in supervisory control. *Human Factors : The Journal of Human Factors and Ergonomics Society*, *37*(1), 5–19.

Scheck, W. (2017). Lawrence sperry : Genius on autopilot. HistoryNet. Retrieved 2023-10-02. Récupéré de https://www.historynet.com/lawrence-sperry-genius-autopilot/

Scheutz, M. et Schermerhorn, P. (2011). Toward humanlike task-based dialogue processing for human robot interaction. *AI Magazine*, *32*(4), 77–84. http://dx.doi.org/10.1609/aimag.v32i4.2381

Schoppek, W. et Boehm-Davis, D. A. (2004). Opportunities and challenges of modeling user behavior in complex real world tasks. *MMI interaktiv*, *7*, 47–60.

Serratosa, F. (2021). Redefining the graph edit distance. *SN Computer Science*, *2*(5), 438. http://dx.doi.org/10.1007/s42979-021-00792-5

Sheng, Y., Chen, X., Mo, H., Chen, X. et Zhang, Y. (2020). An ontology for decision-making support in air traffic management. Dans *Artificial Intelligence in China*, volume 572 de *Lecture Notes in Electrical Engineering*, 458–466. http://dx.doi.org/10.1007/978-981-15-0187-6_55

Simon, H. A. (1996). *The Sciences of the Artificial* (3rd ed. éd.). Cambridge, MA : The MIT Press.

Smart, P. R., Scutt, T., Sycara, K. et Shadbolt, N. R. (2016). Integrating act-r cognitive models with the unity game engine. Dans *Proceedings of the 2016 International Conference on Cognitive Modeling*, Hershey, Pennsylvania, USA. GI Global.

Somers, S. et West, R. (2013). Steering control in a flight simulator using act-r. Dans *Proceedings of the International Conference on Cognitive Modeling*, 1–6. Institute of cognitive science.

Squire, L. R. (1992). Declarative and nondeclarative memory : multiple brain systems supporting learning. *Journal of Cognitive Neuroscience*, *4*(3), 232–243.

Stefanidis, D., Christodoulou, C. et Symeonidis, M. e. a. (2020). The icarus ontology : A general aviation ontology developed using a multi-layer approach. Dans *Proceedings of the 10th International Conference on Web Intelligence, Mining and Semantics (WIMS 2020)*, 21–3. Association for Computing Machinery. http://dx.doi.org/10.1145/3405962.3405983

Stewart, T. C., Choo, F.-X. et Eliasmith, C. (2012). Spaun : A perception-cognition-action model using spiking neurons. *Cognitive Science*, *34*. Récupéré de https://api.semanticscholar.org/CorpusID:5740116

Stewart, T. C. et Eliasmith, C. (2014). Large-scale synthesis of functional spiking neural circuits. *Proceedings of the IEEE*, *102*(5), 881–898. http://dx.doi.org/10.1109/JPROC.2014.2306061

Strube, G. (2001). Cognitive modeling : Research logic in cognitive science. In N. J. Smelser et P. B. Baltes (dir.), *International Encyclopedia of the Social & Behavioral Sciences* 2124–2128. Pergamon

Sun, R. (2002). *Duality of the Mind : A Bottom-Up Approach Toward Cognition*. Lawrence Erlbaum Associates Publishers.

Sun, R. (2004). Desiderata for cognitive architectures. *Philosophical Psychology*, *17*(3), 341–373.

Sun, R. (2006). The clarion cognitive architecture : Extending cognitive modeling to social simulation. In R. Sun (dir.), *Cognition and Multi-Agent Interaction : From Cognitive Modeling to Social Simulation* 79–99. Cambridge University Press.

Sun, R. (2016). *Anatomy of the Mind : Exploring Psychological Mechanisms and Processes with the Clarion Cognitive Architecture*. Oxford University Press. http://dx.doi.org/10.1093/acprof:oso/9780199794553.001.0001

Sun, R., Merrill, E. et Peterson, T. (2001). From implicit skills to explicit knowledge : A bottom-up model of skill learning. *Cognitive Science*, *25*(2), 203–244. http://dx.doi.org/10.1207/s15516709cog2502_2

Taatgen, N. A. (1999). Cognitive modelling : A new look at individual differences. *Dutch Journal of Psychology*, *54*(4), 167–176.

Taatgen, N. A., Huss, D. et Anderson, J. R. (2008). The acquisition of robust and flexible cognitive skills. *Journal of Experimental Psychology : General*, *137*(3), 548–565. http://dx.doi.org/10.1037/0096-3445.137.3.548

Tamkodjou Tchio, G. C., Courtemanche, M. A., Tato, A., Nkambou, R. et Psyché, V. (2023). Integrating an ontological reference model of piloting procedures in act-r cognitive architecture to simulate piloting tasks. Dans C. Frasson, P. Mylonas, et C. Troussas (dir.). *Augmented Intelligence and Intelligent Tutoring Systems*, volume 13891 de *Lecture Notes in Computer Science*, 199–213., Cham. Springer. http://dx.doi.org/10.1007/978-3-031-32883-1_16

Tamkodjou Tchio, G. C., Nkambou, R., Psyché, V. et Nyamen Tato, A. A. (2025). Enhancing pilot training and decision-making using ontologies : A cognitive assistance approach. Dans S. Graf et A. Markos (dir.). *Generative Systems and Intelligent Tutoring Systems. ITS 2025*, volume 15723 de *Lecture Notes in Computer Science*. Springer, Cham. http://dx.doi.org/10.1007/978-3-031-98281-1_5. Récupéré de https://doi.org/10.1007/978-3-031-98281-1_5

Tamkodjou Tchio, G. C., Nkambou, R., Tato, A. A. N. et Psyché, V. (2024a). Towards cognitive coaching in aircraft piloting tasks : Building an act-r synthetic pilot integrating an ontological reference model to assist the pilot and manage deviations. Dans A. Sifaleras et F. Lin (dir.). *Generative Intelligence and Intelligent Tutoring Systems*, volume 14798 de *Lecture Notes in Computer Science*, 183–194. Springer, Cham. http://dx.doi.org/10.1007/978-3-031-63028-6_16

Tamkodjou Tchio, G. C., Nkambou, R., Tato Nyamen, A. A. et Psyché, V. (2024b). Handling of abnormal aircraft takeoff procedures : Cognitive modeling of an act-r synthetic pilot integrating an ontological reference model. Dans P. Mylonas, D. Kardaras, et J. Caro (dir.). *Novel and Intelligent Digital Systems : Proceedings of the 4th International Conference (NiDS 2024)*, volume 1170 de *Lecture Notes in Networks and Systems*, Cham. Springer. http://dx.doi.org/10.1007/978-3-031-73344-4_38

Tato, A., Nkambou, R. et Nana Tato, G. J. (2022). Towards adaptive coaching in piloting tasks : Learning pilots' behavioral profiles from flight data. Dans *Intelligent Tutoring Systems : 18th International Conference, ITS 2022, Bucharest, Romania, June 29–July 1, 2022, Proceedings*, 105–114. Springer.

Tato, A., Nkambou, R. et Nana Tato, G. J. (2023). Automatic learning of piloting behavior from flight data. Dans *Augmented Intelligence and Intelligent Tutoring Systems : 19th International Conference, ITS 2023, Corfu, Greece, June 2–5, 2023, Proceedings*, volume 13891 de *Lecture Notes in Computer Science*, 541–552. Springer.

Tea, P. et Soulé, G. (2024). Apport et utilisation de l'ia dans le domaine de l'aéronautique. *Annales des Mines - Réalités industrielles*, *242*(2), 128–132. http://dx.doi.org/10.3917/rindu1.242.0128. Récupéré de https://doi.org/10.3917/rindu1.242.0128

Thórisson, K. et Helgasson, H. (2012). Cognitive architectures and autonomy : a comparative review. *Journal of Artificial General Intelligence*, *3*(2), 1–30.

Touzet, C. (1992). *LES RESEAUX DE NEURONES ARTIFICIELS, INTRODUCTION AU CONNEXIONNISME*. Collection de l'EERIE. EC2. Disponible sur HAL : https://hal.archives-ouvertes.fr/hal-01338010.

Tsotsos, J. K. (2011). *A Computational Perspective on Visual Attention*. Cambridge : MIT Press.

Tsotsos, J. K. et Kruijne, W. (2014). Cognitive programs : software for attention's executive. *Frontiers in Psychology*, *5*, 1–16. http://dx.doi.org/10.3389/fpsyg.2014.01255

Université TÉLUQ (2019). Définition de l'intelligence artificielle. © Université TÉLUQ, 2019. Tous droits réservés. Récupéré le 2025-03-31 de https://clom-motsia.teluq.ca/module-1/definition-ia/

V-Prep (2017). A320 engine failure after takeoff training. https://www.youtube.com/watch?v=KyWBCiQYRVM.

V-Prep (2018a). A320 predictive and reactive windshear. https://www.youtube.com/watch?v=Z-ptBJD326M&t=65s.

V-Prep (2018b). A320 stall recovery procedure. https://www.youtube.com/watch?v=8YFNmXmYijI.

Vabba, A., Panasiti, M. S., Scattolin, M., Spitaleri, M. et Porciello, G. (2023). The thermoception task : a thermal imaging-based procedure for measuring awareness of changes in peripheral body temperature. *Journal of Neurophysiology*. Dimensions ID : pub.1162938722, http://dx.doi.org/10.1152/jn.00014.2023

Van, T. (2014). George the autopilot. Historic Wings. Accessed 2025-02-26.

Varela, F. J., Thompson, E. et Rosch, E. (1991). *The Embodied Mind : Cognitive Science and Human Experience*. The MIT Press.

Vernon, D., Metta, G. et Sandini, G. (2007). A survey of artificial cognitive systems : implications for the autonomous development of mental capabilities in computational agents. *IEEE Transactions on Evolutionary Computation*, 1–30.

W3C (2012). Owl 2 web ontology language : Structural specification and functional-style syntax (second edition). https://www.w3.org/TR/owl2-overview.

Wickens, C. D., Helton, W. S., Hollands, J. G. et Banbury, S. (2021). *Engineering Psychology and Human Performance* (5th éd.). Routledge. http://dx.doi.org/10.4324/9781032011738

Wickens, C. D., Mavor, A. S., Parasuraman, R. et McGee, J. P. (1998). *The Future of Air Traffic Control : Human Operators and Automation*. National Academies Press. http://dx.doi.org/10.17226/6018

Wiener, E. L. et Curry, R. E. (1980). Flight-deck automation : promises and problems. *Ergonomics*, *23*(10), 995–1011. http://dx.doi.org/10.1080/00140138008924809

Winston, P. H. (1992). *Artificial Intelligence* (3 éd.), volume 0 de *A-W Series in Computer Science*. Addison-Wesley Publishing Company.

Wu, Y., Ebrahimipour, V. et Yacout, S. (2014). Ontology-based modeling of aircraft to support maintenance management system. Dans Y. Guan et H. Liao (dir.). *Proceedings of the 2014 Industrial and Systems Engineering Research Conference*.

Yokoyama, D. et Harada, A. (2021). Verification of flight data mimicked by flight simulator software, x-plane. *AEROSPACE TECHNOLOGY JAPAN*, *THE JAPAN SOCIETY FOR AERONAUTICAL AND SPACE SCIENCES*, *20*, 154–165. http://dx.doi.org/10.2322/astj.20.154

Yousefzadeh Aghdam, M., Kamel Tabbakh, S. R., Mahdavi Chabok, S. J. et Kheyrabadi, M. (2021). Ontology generation for flight safety messages in air traffic management. *Journal of Big Data*, *8*(1), 1–21. http://dx.doi.org/10.1186/s40537-021-00449-3

Zhang, Z., Russwinkel, N. et Prezenski, S. (2018). Modeling individual strategies in dynamic decision-making with act-r : A task toward decision-making assistance in hci. *Procedia Computer Science*, *145*, 668–674. http://dx.doi.org/10.1016/j.procs.2018.11.064

Zhao, M., Jia, W., Jennings, S. *et al.* (2024). Monitoring pilot trainees' cognitive control under a simulator-based training process with eeg microstate analysis. *Scientific Reports*, *14*, 24632. http://dx.doi.org/10.1038/s41598-024-76046-0