HAL
open science

Conception d'un système tutoriel intelligent orienté stylet pour l'apprentissage de la géométrie basé sur une interprétation à la volée de la production manuscrite de figures

Omar Krichen

▶ **To cite this version:**

Omar Krichen. Conception d'un système tutoriel intelligent orienté stylet pour l'apprentissage de la géométrie basé sur une interprétation à la volée de la production manuscrite de figures. Vision par ordinateur et reconnaissance de formes [cs.CV]. INSA de Rennes, 2020. Français. NNT : 2020ISAR0006 . tel-03192038

HAL Id: tel-03192038

https://theses.hal.science/tel-03192038v1

Submitted on 7 Apr 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT DE

L'Institut National des Sciences Appliquées de Rennes

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies
de l'Information et de la Communication
Spécialité : *Informatique*

Par
«Krichen Omar »

« Conception d'un système tutoriel intelligent orienté stylet pour l'apprentissage de la géométrie basé sur une interprétation à la volée de la production manuscrite de figures »

Thèse présentée et soutenue à « Rennes », le « 4/12/2020 »
Unité de recherche : IRISA - UMR6074
Thèse N° : 20ISAR 25 / D20 - 25

Rapporteurs avant soutenance :

Stéphanie Jean-Daubias	Professeure des écoles, université Claude-Bernard Lyon 1
Jean-Yves Ramel	Professeur des écoles, université de Tours

Composition du Jury :

Présidente :	Dominique Py	Professeure des universités, université du Mans
Examinateurs :	Stéphanie Jean-Daubias	Professeure des universités, université Claude-Bernard Lyon 1
	Jean-Yves Ramel	Professeur des universités, université de Tours
	Véronique Eglin	Professeure des universités, INSA Lyon
	Harold Mouchère	Professeur des universités, université de Nantes
Dir. de thèse :	Éric Anquetil	Professeur des universités, INSA Rennes

Invité(s) :

Co-encadrante de thèse :	Nathalie Girard	Maître de conférences, université Rennes 1
	Éric Jamet	Professeur des universités, université Rennes 2

Remerciements

Je voudrais d'abord exprimer ma reconnaissance à l'ensemble des membres du jury pour l'intérêt qu'ils ont porté à cette thèse. Je remercie Stéphanie Jean-Daubias et Jean-Yves Ramel d'avoir bien voulu rapporter ce manuscrit. Merci aussi à Véronique Eglin, Harold Mouchère, et Dominique Py d'avoir accepté d'être les examinateurs de ma soutenance. Enfin, merci à Eric Jamet, pour sa participation à ce jury en tant qu'invité, et pour sa collaboration à ce travail de thèse.

Une immense gratitude va naturellement à mes encadrants, Eric Anquetil et Nathalie Girard. Au delà du temps précieux qu'ils m'ont consacré, je leur suis reconnaissant à la fois du savoir qu'ils m'ont transmis, de la liberté d'exploration qu'ils m'ont accordée, et du soutien tant scientifique que moral qu'ils m'ont apporté. Je pense aussi aux soirées et aux week-ends sacrifiés à la relecture de mes articles, et notamment de ce manuscrit, et je me rends compte de la chance que j'ai eu de les avoir comme encadrants. Encore une fois, merci. Au passage, un message pour mon hypothétique futur(e) doctorant(e) : désolé, pour moi le week-end, c'est sacré, je ne rate mon foot pour rien au monde, sauf peut-être pour une partie de ton salaire.

Un grand, très grand merci à mes camarades ingénieurs qui m'ont apporté une aide ô combien précieuse durant ces années : Mickaël Renault, Simon Corbillé, et Morgane Carry pour la géométrie, et Richard Lagrange pour l'adaptation du système au domaine de l'architecture. Je leur suis particulièrement reconnaissant pour leur travail sur l'IHM et l'ergonomie du système développé. C'est en grande partie en grâce à eux que cette thèse a pu être réalisée sous de bonnes conditions.

Je voudrais aussi exprimer ma gratitude envers nos partenaires du Loustic/LP3C, Eric Jamet (encore un merci !), Maxime Robert, Coraline Bourges, Elisa Lienhart, et Tiphaine Colliot, pour leur collaboration précieuse, et notamment pour l'organisation des expérimentations en classes du système. J'en profite pour remercier aussi les partenaires de l'académie de Rennes, les enseignants et élèves des collèges La Roche aux Fées et Paul Sabillot d'avoir participé aux expérimentations et d'avoir donné un sens à ce travail.

Une pensée chaleureuse à tous les membres de l'équipe IntuiDoc pour la bonne ambiance, les cafés, les sorties, et surtout d'avoir opté un temps pour la coinche (ou comme

je l'appelle, la belote tunisienne) en lieu et place de la belote classique. Je considère cela comme une victoire personnelle, un de mes plus grands succès dans ce monde ici-bas.

Un merci infiniment grand à toi Tata Hikma si tu lis ces pages. Je ne serai sûrement pas arrivé jusque là sans ton aide. Je serais peut-être entrain de repasser le concours de prépa une énième fois à l'heure qu'il est si tu n'avais pas été là.

Vu qu'aucun de mes amis ne lira ce manuscrit, je vais être rapide : merci pour tout. Si tu lis ces lignes et que tu es vexé(e), que tu penses qu'il faudrait quand même développer un peu plus, appelles-moi pour me réprimander et je t'expliquerai combien tu comptes pour moi.

Par contre, mes parents vont sûrement essayer de le lire, avant de s'endormir à la page 20. Fatma, Zyed, je vous dois tout. A mesure que les années passent, et que ma compréhension de la vie s'accroît, je me rends compte de l'immense chance que j'ai de vous avoir. Après avoir cogité longtemps, je n'arrive pas à trouver une meilleure expression que celle que je vous disais quand j'étais un gamin de six ans : je vous aime grand comme l'univers.

Table des matières

Introduction **11**

I État de l'art **21**

1 Fondements pédagogiques de la thèse **22**
1.1 Apprentissage actif et dessin génératif 22
1.2 L'intérêt du dessin libre et de la manipulation intuitive 23
1.3 L'intérêt de la supervision et du feedback immédiat 25
1.4 Bilan 26

2 Interprétation à la volée de tracés manuscrits **27**
2.1 Définitions générales 27
2.1.1 Documents structurés 27
2.1.2 Composition de documents structurés 28
2.1.3 Interprétation en ligne de documents manuscrits 30
2.2 Approches pour l'interprétation en-ligne de diagrammes manuscrits 32
2.3 État de l'art des approches structurelles 34
2.3.1 Grammaires formelles 34
2.3.2 Grammaires à base d'opérateurs 36
2.3.3 Grammaires à base de fonctions 38
2.3.4 Grammaires à base de graphes 41
2.3.5 Bilan 42

3 État de l'art sur les systèmes tutoriels intelligents **44**
3.1 Définitions générales 44
3.1.1 Qu'est ce qu'un système tutoriel intelligent ? 44
3.1.2 Historique des STI 45
3.1.3 L'architecture en quatre composants des STI 45
3.2 Approches et philosophies des STI 47

3.2.1 Tuteurs à base de règles 48
3.2.2 Tuteurs à base de contraintes 51
3.2.3 Tuteurs à base de traçage d'exemples 53
3.2.4 Approches basées sur les données 55
3.3 Systèmes tutoriels dédiés à l'apprentissage de la géométrie 56
3.4 Systèmes tutoriels basés sur du dessin 59
3.5 Discussion et bilan 62
3.6 Préambule de la suite du manuscrit 63

II Interprétation à la volée de schémas géométriques 65

4 GMC-PC : Grammaire Multi-ensembles à Contraintes Pilotée par le Contexte 66
4.1 Définitions et principes généraux 67
4.2 Modélisation de la connaissance par les règles de productions 68
4.2.1 Étape de vérification : le bloc de préconditions 68
4.2.2 Étape de reconnaissance : le bloc de contraintes 70
4.2.3 Étape de prédiction : les postconditions 70
4.3 Processus d'analyse 71
4.3.1 Analyse guidée par le contexte 72
4.3.2 Évaluation d'une production 72
4.3.3 Construction de l'arbre d'analyse et stratégies de recherche 74
4.4 Limites de GMC-PC et extension du formalisme 77

5 Grammaire Multi-ensembles à Contraintes Hiérarchique et Guidée par le Contexte 79
5.1 Modélisation de la connaissance géométrique a priori 80
5.1.1 Graphe sémantique de connexion et polygones 81
5.1.2 Production de segments 84
5.2 Hiérarchisation des contextes 86
5.3 Hiérarchisation des règles de production 90
5.4 Analyseur Contextuel Sémantique 93
5.4.1 Génération automatique à partir de la définition de l'utilisateur . . 93
5.4.2 Processus d'analyse 95

5.4.3 Stratégie d'optimisation 97
5.5 Générécité de l'approche : application à la composition de plans architecturaux 99
5.6 Bilan 101

6 Expérimentations et résultats relatifs à l'extension du formalisme 102
6.1 Impact de la hiérarchisation des contextes 103
6.2 Impact de la hiérarchisation des règles de production 104
6.3 Impact de l'analyseur contextuel sémantique 106
6.4 Discussion 108

III Tutorat et supervision d'exercices de construction en géométrie 109

7 Principes généraux du tuteur 110
7.1 Génération d'exercices à partir d'un exemple solution 111
7.2 Supervision et évaluation : modélisation de l'élève 112
7.3 Synthèse de stratégies de guidage : modélisation de l'expert 113
7.4 Bilan 114

8 Modélisation de la connaissance déclarative spécifique au problème et modélisation de l'élève 116
8.1 Génération du modèle à partir d'exemple 116
8.2 Construction du graphe de connaissance 117
8.3 Modélisation de l'état de résolution de l'élève 120
8.3.1 Cas de base : vérification d'élément labellisé par l'élève 121
8.3.2 Interprétation sémantique et mise en correspondance 121
8.4 Bilan 126

9 Module expert et synthèse de stratégies de résolution 127
9.1 Le domaine de planification du module expert 128
9.2 Les actions de dessin du module expert 128
9.3 Décomposition du problème de planification en sous-problèmes 132
9.4 Algorithme de planification 133
9.5 Bilan 137

10 Expérimentations et résultats relatifs à la performance du moteur de supervision 139
10.1 Protocole d'évaluation 139
10.2 Évaluation de la performance du module de l'apprenant 140
10.3 Performance du module expert 141
10.4 Bilan 142

IV Prototypage et expérimentation en classe 145

11 Prototypage d'IntuiGeo 146
11.1 Interface et fonctions d'éditions 147
11.1.1 Interface 147
11.1.2 Fonctions d'éditions 148
11.2 Outils virtuels et conception centrée utilisateur 151
11.3 Feedback : visuels, de correction, et de guidage 154
11.3.1 Feedback visuels 154
11.3.2 Feedback de correction 155
11.3.3 Feedback de guidage 157
11.4 Possibilités d'évaluation pour l'enseignant 159
11.5 Bilan 161

12 Impact pédagogique d'IntuiGeo 162
12.1 Protocole de l'étude 163
12.2 Procédure d'évaluation des performances 164
12.2.1 Évaluation du niveau initial 164
12.2.2 Phase d'entraînement 164
12.2.3 Phase d'apprentissage 165
12.2.4 Phase de transfert 165
12.3 Résultats 165
12.3.1 Résultats sur l'évaluation du niveau initial 166
12.3.2 Résultats sur la phase d'apprentissage 166
12.3.3 Résultats sur la phase de transfert 167
12.3.4 Mesures subjectives 168
12.4 Bilan 169

Conclusion et perspectives **171**

Publications de l'auteur **175**

Bibliographie **177**

Introduction

Contexte de la thèse : le projet ACTIF

Cette thèse s'inscrit dans le projet ACTIF (Apprentissage et Collaboration sur Tablettes, Interactions et Feedback), soutenu par le ministère de l'éducation nationale français, sélectionné sur l'appel à projet e-FRAN (espaces de Formation, de Recherche et d'Animation du Numérique) émis dans le cadre du Programme d'investissement d'avenir et porté par le GIP-FAR. L'objectif principal est de profiter des avancées du numérique, et notamment la démocratisation des tablettes orientées stylet dans les classes, pour stimuler l'apprentissage actif et collaboratif des élèves. Ce projet est pluridisciplinaire et a permis la collaboration de plusieurs unités de recherche dont les travaux portent sur l'informatique, l'intelligence artificielle, la psychologie cognitive et sociale, et l'ergonomie :

— L'équipe IntuiDoc du laboratoire IRISA a apporté son savoir faire en termes de reconnaissance de documents manuscrits et d'interprétation de tracés en-ligne, et a développé le système d'e-apprentissage ;
— Le LP3C, avec son expertise sur les approches d'apprentissage, a apporté sa méthodologie scientifique pour le traitement des données expérimentales ;
— La plate-forme Loustic a géré la coordination de la conception centrée utilisateur : l'interaction avec les enseignants et les élèves, ainsi que les expérimentations en classes, etc.

Le projet ACTIF se décline en trois volets : le premier porte sur la production de schémas et de feedback en temps-réel, le deuxième sur l'enseignement par les pairs, et le troisième sur le travail collaboratif. Ce travail de thèse s'inscrit dans le cadre du premier volet en se focalisant sur l'apprentissage de la géométrie au collège.

IntuiGeo, tuteur interactif pour l'apprentissage de la géométrie

L'objectif de ces travaux de thèse est de concevoir un tuteur interactif pour l'apprentissage de la géométrie que nous avons dénommé IntuiGeo (Intuitive Geometry). L'originalité de ce tuteur est son aspect intuitif. En effet, l'utilisation des tablettes avec stylet permet de simuler l'approche papier/crayon, afin de rendre le support numérique transparent à l'élève, et d'atteindre ainsi une transférabilité de l'apprentissage entre les supports numérique et papier. De plus, l'usage de la tablette permet de proposer un parcours personnalisé à chaque élève. Ainsi, le tuteur IntuiGeo, conçu et développé pendant cette thèse, est un système d'e-apprentissage sur tablette qui se base sur deux grands principes :

1. La capacité de reconnaître et d'analyser, à la volée, les productions de l'élève ;
2. La capacité de superviser, en temps-réel, les stratégies de résolution d'un exercice de construction en géométrie, afin de générer des feedbacks de correction et de guidage personnalisés et adaptés à l'état d'avancement de l'élève.

De ces deux principes, nous pouvons dégager les grands axes de cette thèse, correspondant à deux domaines de recherche, la reconnaissance à la volée de tracés manuscrits d'un coté, et la modélisation de la connaissance et de l'apprenant pour la supervision de stratégies de résolution d'exercices de l'autre. La figure 1 présente l'utilisation du tuteur IntuiGeo par un élève en classe, nous pouvons voir l'élève résoudre un problème de construction en

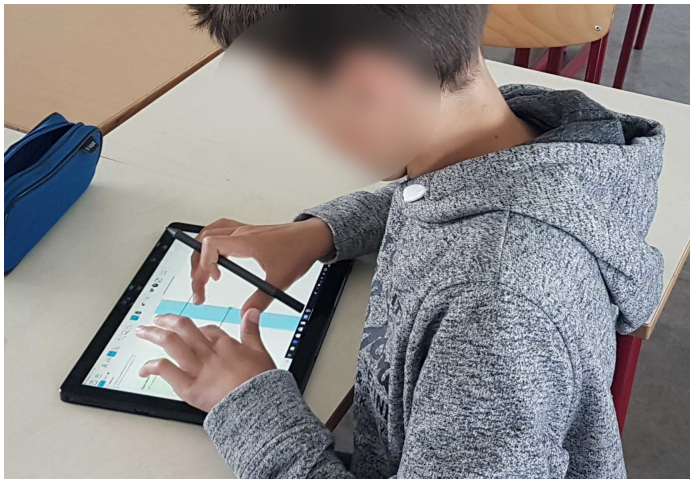
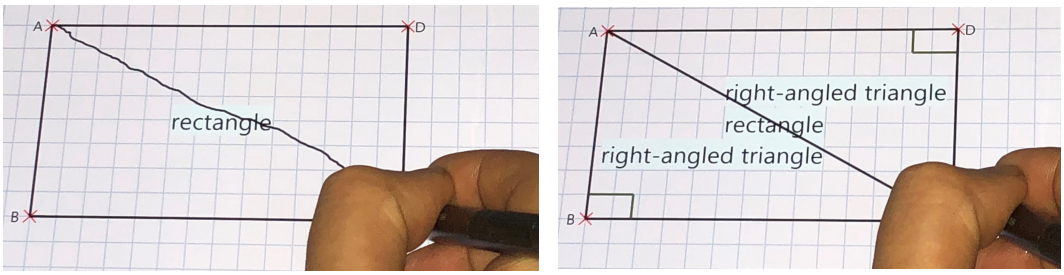


FIGURE 1 – Simulation de la condition papier-crayon dans IntuiGeo : l'élève manipule avec son doigt une règle virtuelle et trace (les segments, les arcs, etc) avec le stylet

utilisant une règle virtuelle *via* un contrôle tactile, alors qu'il a utilisé son stylet pour dessiner les segments déjà reconnus. Suite à cette présentation du contexte général, intéressons nous maintenant aux axes principaux de notre travail explicités plus haut.

Premier axe : reconnaissance à la volée de tracés manuscrits

Ce premier axe de la thèse porte sur la reconnaissance à la volée de la production de l'élève. Nous entendons par production la construction d'une figure géométrique étant donné une consigne / exercice. Notre objectif est de permettre à l'élève de dessiner librement, en simulant sur une tablette stylet la condition traditionnelle de réalisation de figures, *i.e.* la production sur papier avec un crayon et des outils de dessin tels qu'un compas, une équerre, etc. La plupart des logiciels de géométrie dynamique existants et utilisés dans les écoles se basent sur une approche glisser-déposer (en utilisant des boutons et des menus), et s'éloignent de la condition traditionnelle papier/crayon. Cela implique une moins bonne transférabilité de l'apprentissage, et un temps de prise en main du logiciel plus long. Nous y reviendrons plus tard en détail. Pour permettre le dessin libre de l'élève, nous nous intéressons à *l'interprétation à la volée de documents manuscrits structurés*, illustrée dans la figure 2. Un des challenges qui incombent à ce type d'interprétation est de gérer la complexité du processus d'analyse et de reconnaissance en prenant en compte la contrainte de l'interaction temps-réel entre l'utilisateur et le système. Nous nous basons dans cette thèse sur un formalisme grammatical bidimensionnel GMC-PC (Grammaires Mutli-ensembles à Contraintes Pilotées par le Contexte) [MA09]. La connaissance a priori du domaine est modélisée par les règles de production et l'analyseur associé à la grammaire est capable d'interpréter les tracés manuscrits de l'utilisateur. Ce formalisme est générique, *i.e.* il peut être utilisé pour tout type de documents structurés et a été adapté



Tracé dessiné — Interprétation et retour visuel

FIGURE 2 – Processus d'interprétation à la volée

à plusieurs domaines tels que le dessin de circuits électriques ou les plans d'architecture [Gho12]. Cependant, l'analyseur atteignait ses limites lorsque nous voulions introduire de la sémantique au document, par exemple la notion de pièce sur un plan d'architecture, ou de figures composées sur un dessin de géométrie. Notre contribution sur cet axe consiste à étendre et enrichir ce formalisme pour contrôler la complexité du processus d'analyse, tout en préservant l'aspect générique de la grammaire. Nous démontrerons nos contributions en validant la portée de l'extension du formalisme aussi bien au domaine de la production de schémas géométriques que de la composition de plans architecturaux. Nous avons conçu et développé le moteur de reconnaissance 2D d'IntuiGeo en nous appuyant sur cette approche.

Deuxième axe : supervision de stratégies de résolution en temps-réel

L'aspect *tutoriel* de notre système vient de sa capacité à superviser, en temps réel, les stratégies de résolution choisies par l'élève dans le contexte d'un problème de construction. Nous nous inspirons de la littérature des systèmes tutoriels intelligents (Intelligent Tutoring Systems), pour concevoir un moteur de supervision qui remplit trois rôles principaux :

— La génération automatique d'un problème de construction, à partir d'un exemple solution dessiné par l'enseignant : *module "auteur" de l'application.*
L'enseignant dessine la solution sur l'interface. Sa production est interprétée par le moteur de reconnaissance 2D et induit la génération d'un modèle du problème, sous la forme d'un graphe de connaissance, représentant les éléments géométriques construits par l'enseignant, et les contraintes qui les lient. Ce graphe modélise donc la connaissance spécifique au problème à résoudre, et sera la base de l'évaluation des actions de l'élève par le tuteur ;

— La supervision et l'évaluation en temps-réel de l'état de résolution de l'élève : *module "apprenant" de l'application.*
Au delà de la reconnaissance des tracés de l'élève par le moteur de reconnaissance, une interprétation sémantique des actions de l'élève est nécessaire pour comprendre l'état de son avancement dans la résolution de l'exercice. Pour ce faire, nous définissons un module de l'apprenant, responsable de l'évaluation et du suivi de l'élève dans sa résolution. A chaque nouvelle action de l'élève, le tracé est reconnu et l'élé-

ment interprété est mis en correspondance avec le modèle de l'exercice (le graphe de connaissance). Cela permet un suivi en temps-réel des stratégies de résolution de l'élève et la génération de feedbacks (retours) de correction adaptés ;

— La synthèse de stratégie pour la résolution d'un exercice généré : *module "expert" de l'application.*
  Pour pouvoir superviser la réalisation de l'élève et l'aider dans sa production, le tuteur doit être capable de résoudre l'exercice. Pour ce faire, nous définissons un module expert, basé sur un environnement dynamique de planification adapté au domaine de la géométrie. L'objectif pour ce module expert est de trouver, à partir de n'importe quel état de résolution de l'exercice de l'élève, un plan solution qui sera traduit en feedbacks de guidage sur les prochaines étapes à réaliser.

Après avoir présenté les deux axes de travail représentant les deux thématiques scientifiques associées à cette thèse, nous nous intéressons maintenant aux différentes typologies de feedback qu'IntuiGeo délivre à l'élève pendant la résolution d'un problème de géométrie, ainsi qu'aux outils virtuels réalistes conçus pour simuler les outils traditionnels.

Feedback : visuel, correctif, ou de guidage

L'objectif du tuteur intelligent est l'amélioration de la performance des élèves. Pour ce faire, il faut atteindre un objectif précis : la transférabilité de l'apprentissage entre les deux supports : papier/crayon (traditionnel) et numérique (IntuiGeo). Autrement, dit, nous souhaitons que les compétences qu'a acquises l'élève en utilisant IntuiGeo puissent être les mêmes que celles qu'il aurait acquises dans la condition traditionnelle (papier/crayon). Pour cela différentes typologies de feedback sont intégrées à IntuiGeo :

— Feedback visuel et outils virtuels :
  Puisque l'élève peut dessiner à main levée (à l'aide d'un stylet numérique), les tracés reconnus sont alors automatiquement reconstruits sous leur forme "idéale" (*cf.* figure 2). Ce feedback visuel a une double fonction : permettre une plus grande liberté d'édition à l'élève, et signifier à l'utilisateur que le système a bien reconnu son geste manuscrit. Nous proposons aussi un ensemble d'outils virtuels (règle, compas, rapporteur, équerre) dont le paradigme d'utilisation se rapproche des vrais outils utilisés en classe. En effet, nous profitons de l'interaction stylet-doigts qu'offrent les tablettes orientées stylet pour la manipulation des outils. Ainsi, le dessin se fait avec le stylet tandis que les outils se manipulent avec les doigts. Les résultats de nos

expériences en classes démontrent, entre autres, que ces outils virtuels, manipulés de façon très similaire aux outils réels, aident les élèves à maîtriser les outils réels lorsqu'ils reviennent à la condition papier/crayon.

— Feedback de correction et de guidage :
  Un des points les plus importants que peut apporter IntuiGeo est la capacité d'analyser à la volée et en temps réel la stratégie de résolution de l'élève et de générer un feedback adapté à la situation. La combinaison de notre moteur de reconnaissance de tracés et notre moteur de supervision de stratégies permet la génération de deux types de feedback :
  — Feedback correctif : puisque le tuteur a une connaissance de l'avancement de l'élève, il est capable de générer une traduction visuelle ou textuelle de cet état : les étapes réalisées et les fautes qu'il faut corriger (*e.g.* longueur de segment, perpendicularité, parallélisme, etc.). Ces feedbacks peuvent être générés au fil de la composition de l'élève, ou être fournis à la demande, selon les stratégies pédagogiques envisagées.
  — Feedback de guidage : Quand l'élève est bloqué, il peut demander de l'aide. Le moteur de supervision, connaissant l'état de résolution de l'exercice, essaie alors de trouver un plan de solution à partir de la stratégie de l'enfant. Une partie de ce plan est ensuite affichée à l'élève sous forme de piste de réflexion pour sortir de ce blocage. On distingue deux types de feedback de guidage : un guidage dit "en avant" et qui indique la prochaine étape à réaliser, et un guidage dit "en arrière" si l'élève est dans une impasse et qu'il n'y a pas de résolution possible.

Notre système étant dédié à l'apprentissage de la géométrie au collège, il est indispensable qu'il soit facile à utiliser, intuitif, et donc adapté aux besoins des élèves et des enseignants. Pour ce faire, nous avons suivi un processus de conception centré utilisateur (CCU). Ce processus consiste à impliquer les utilisateurs du produit final dans la conception du système, pour le faire évoluer de façon itérative, en prenant en compte leurs retours collectés lors de phases de tests régulières [AMP04]. Les travaux de collaboration avec les LP3C, le LOUSTIC et les partenaires académiques s'inscrivent dans ce processus.

Conception centrée utilisateur et impact pédagogique

Le partenariat avec nos collègues de l'université Rennes 2 (Loustic, LP3C) et l'académie de Rennes nous a permis de nouer des contacts avec plusieurs enseignants de collèges

bretons, qui se sont portés volontaires pour accompagner notre processus de développement du tuteur et nous ont permis de réaliser des expérimentations en classe.Tout d'abord, les premières expérimentations avaient comme objectif de valider l'ergonomie et l'utilisabilité d'IntuiGeo. Ces expérimentations ont permis, entre autres, de faire évoluer les outils virtuels pour qu'ils correspondent au mieux aux outils réels. Ensuite, les expérimentations ont porté sur l'évaluation de l'impact pédagogique d'IntuiGeo en termes d'apprentissage et de transférabilité des connaissances acquises. Nous présenterons dans le chapitre 12 les résultats positifs de l'utilisation de notre tuteur en termes de performance et transférabilité de l'apprentissage.

Plan du manuscrit

Ce manuscrit est organisé comme suit.

Dans la première partie, nous présentons le contexte global et l'état de l'art relatif aux deux axes de travail de cette thèse.

Dans le chapitre 1, nous illustrons les fondements pédagogiques de ce projet. Plus précisément, les concepts généraux d'apprentissage actif, d'apprentissage par le dessin, et l'intérêt du feedback sont présentés. Nous nous intéressons aussi aux logiciels de géométrie dynamique utilisés en classes, en pointant les limites que nous souhaitons dépasser.

Le chapitre 2 porte sur l'interprétation à la volée des documents structurés. Nous étudions les concepts généraux et les différentes techniques et approches. Nous focalisons cette étude sur les méthodes dites *à la volée* en mettant en lumière la complexité de ce type d'approche, surtout dans le contexte d'une interaction temps-réel avec l'utilisateur.

Le chapitre 3 présente la littérature portant sur les systèmes tutoriels intelligents, un domaine à part entière dans la communauté de l'intelligence artificielle. Nous en dégagerons les différentes caractéristiques nécessaires à notre approche pour réussir à concevoir un système interactif capable d'accompagner en temps-réel l'élève dans la résolution de son exercice.

La deuxième partie porte sur nos contributions dans le premier axe de cette thèse : l'interprétation à la volée de schémas géométriques.

Le chapitre 4 introduit le formalisme grammatical GMC-PC existant, dédié à la modélisation et à l'analyse à la volée de tracés manuscrits. Nous mettons en évidence son

expressivité, sa généricité, et ses limites.

Dans le chapitre 5, nous décrivons notre approche pour étendre et enrichir ce formalisme, afin de contrôler la complexité du processus d'analyse, tout en préservant son aspect générique.

Nous présenterons dans **le chapitre 6** les résultats sur ce premier axe, avec des expérimentations sur deux domaines d'application : la construction à main levée de figures géométriques, et la composition de plans d'architecture. Ces résultats correspondent à l'évaluation du moteur de reconnaissance 2D basé sur notre extension du formalisme.

La troisième partie porte sur nos contributions dans le deuxième axe de cette thèse, avec la définition du moteur de supervision, qui est à la base de l'aspect tutoriel de notre système.

Nous présenterons **dans le chapitre 7** les principes généraux de notre approche tutorielle.

Le chapitre 8 portera sur deux modules du moteur de supervision : d'un côté le module "auteur" avec la génération assistée de problèmes de construction en géométrie et la modélisation d'exercices par des graphes de connaissances, de l'autre le module "apprenant" et la modélisation de l'état de résolution de l'élève et l'évaluation de ses actions au fil de l'eau.

Le chapitre 9 portera sur le module expert, défini par un environnement dynamique de planification, modélisant la connaissance du domaine, et la capacité du tuteur à générer des plans solution et des stratégies de guidage de façon dynamique, à partir de n'importe quel état de résolution de l'élève.

Le chapitre 10 présentera les résultats de l'évaluation du moteur de supervision.

La quatrième partie porte sur le prototype que nous avons conçu et développé, et sur les expérimentations du tuteur dans l'écosystème du collège.

Nous présenterons **dans le chapitre 11** l'interface, les outils virtuels, et la typologie des feedbacks en mettant en avant l'influence du processus de conception centrée utilisateur sur l'évolution du prototype.

Dans le chapitre 12, nous nous intéresserons à l'étude d'impact pédagogique sur l'apprentissage, étude effectuée dans des classes de collège en Bretagne. Nous démontrerons que l'utilisation d'IntuiGeo a un impact positif sur la performance des élèves, et que les compétences acquises sur tablettes sont transférables sur le support papier.

Nous enchaînerons enfin par une conclusion et des perspectives, où nous résumerons les contributions présentées, ainsi que les travaux potentiels futurs à réaliser en se basant sur ce travail. Pour être plus précis, ils s'agira de pistes visant à enrichir le parcours pédagogique offert à l'élève par l'outil, par exemple en proposant des exercices ciblés et adaptés aux compétences et lacunes de l'apprenant.

Première partie

État de l'art

Chapitre 1

Fondements pédagogiques de la thèse

Ce chapitre présente les fondements pédagogiques de ce travail. Nous nous intéresserons au concept d'apprentissage actif qui motive l'utilisation de la tablette en classe. Nous étudierons aussi l'existant en termes de logiciels d'apprentissage de la géométrie, qui ont été introduits en classe, avec leurs avantages et leurs limites.

1.1 Apprentissage actif et dessin génératif

L'apprentissage actif, tel que défini dans [FM91], caractérise le fait que les élèves participent dans le processus d'acquisition de la connaissance et ne se contentent pas uniquement d'écouter passivement le cours de l'enseignant. Les activités telles que la lecture, le débat, le dessin, ou la prise de note stimulent la motivation et permettent le développement des aptitudes de l'élève. Dans [FM15], les auteurs estiment que l'apprentissage est actif quand l'élève essaye activement de comprendre le matériel qui lui est proposé en s'impliquant dans un traitement cognitif de l'information. Ce traitement se compose de trois actions :

1. Sélectionner : prêter attention aux informations pertinentes (mémoire sensorielle) ;
2. Organiser : organiser les informations sélectionnées dans des structures cognitives cohérentes (mémoire opérationnelle) ;
3. Intégrer : intégrer ces structures avec des connaissances acquises (mémoire long terme).

Toujours dans [FM15], plusieurs stratégies sont proposées pour stimuler l'apprentissage actif, parmi lesquelles apprendre en dessinant, apprendre en résumant, apprendre en enseignant à un binôme, apprendre en s'expliquant à soi-même, ou apprendre en jouant un

rôle. *Nous nous intéressons ici à l'apprentissage par le dessin, communément appelé dessin génératif (generative drawing).* Le traitement cognitif de l'information dans le contexte de l'apprentissage par le dessin est le suivant :

— Sélectionner : l'élève choisit les éléments à inclure dans son dessin ;
— Organiser : l'élève organise spatialement les éléments dans le dessins ;
— Intégrer : l'élève traduit l'information verbale en information visuelle.

Plus intuitivement, il est clair que la géométrie de construction permet de comprendre et assimiler plus facilement et durablement les concepts mathématiques. Le tout est de profiter de la démocratisation des tablettes orientées stylet dans les classes du collège pour proposer des méthodes intuitives permettant de stimuler cet apprentissage actif par le dessin.

1.2 L'intérêt du dessin libre et de la manipulation intuitive

L'introduction du numérique en classe a commencé dès les années 80 [Hub15], avec pour idée d'introduire l'ordinateur dans le schéma didactique enseignant-élève-savoir. Dans ce contexte, les premiers environnements d'apprentissage humain dédiés aux mathématiques, et plus précisément à la géométrie, ont vu le jour. L'idée était d'enrichir l'expérience de l'élève en classe et de stimuler la compréhension des concepts en facilitant la visualisation, la construction, et la manipulation de figures géométriques. La notion de manipulation correspond à la capacité à sélectionner des figures à partir d'une barre d'outils, et les manipuler (déplacer, agrandir, etc), il est alors question de "géométrie dynamique". De nombreux outils de l'état de l'art se basent sur ce paradigme de géométrie dynamique, nous pouvons citer GeoTouch [Iso+14], SketchGeometry [MW13], Geometer SketchPad [Sch00], ou les plus populaires Cabri-Géomètre [Lab02] et Geogebra [BC15]. La figure 1.1 illustre l'interface et le mode d'interaction de Cabri-Géomètre, assez représentatif des logiciels de géométrie dynamique. En général, pour estimer l'impact d'un outil numérique d'apprentissage de la géométrie, la méthodologie consiste à comparer la performance d'un groupe d'élève utilisant le logiciel à celle d'un groupe de contrôle en condition traditionnelle papier-crayon (nous suivrons cette approche pour évaluer notre système tutoriel). La plupart des études tendent à démontrer l'impact positif qu'ont les logiciels de géométrie dynamique sur la performance de l'élève [SAT10][Cha+14].

S'il est clair que la capacité de manipuler des figures dans l'interface graphique permet

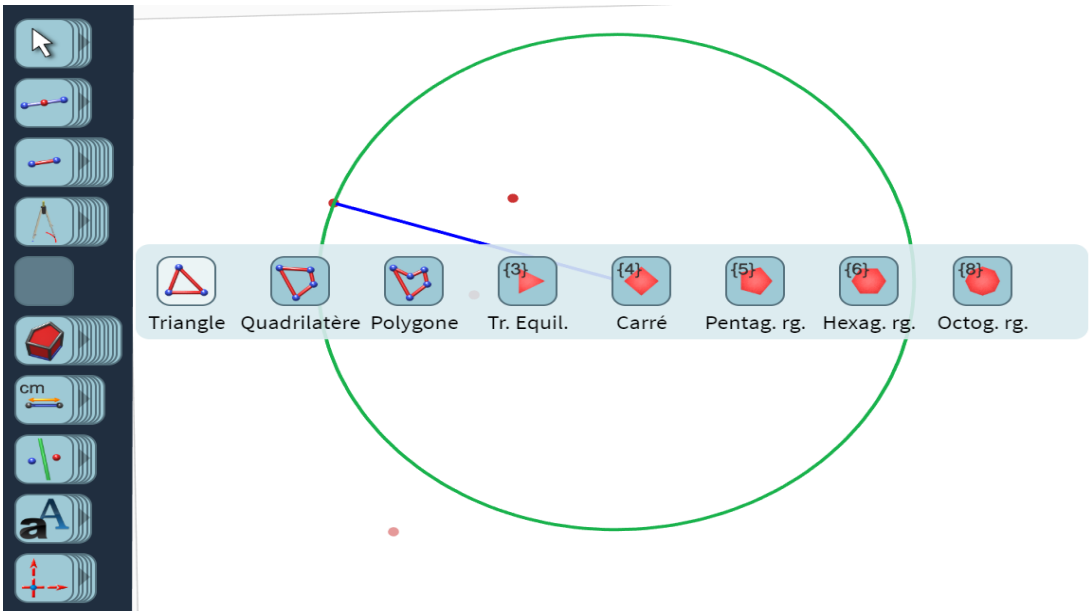


FIGURE 1.1 – Interface de Cabri-Géométrie

de comprendre des concepts géométriques compliqués tels que les transformations géométriques (translation, homothétie, etc), une limite claire est que l'élève perd de la liberté sur la construction de la figure géométrique. Comme illustré dans la figure 1.1, les logiciels de géométrie dynamique se caractérisent par une approche "glisser-déposer" (en utilisant des boutons et des menus). En effet, dans les logiciels de géométrie dynamique, l'élève choisit un objet à dessiner (triangle, quadrilatère...) et le dépose sur la zone de dessin. Il peut ensuite modifier les positions de ses extrémités et observer le changement des propriétés de la figure. L'approche mise en oeuvre s'éloigne donc de la construction traditionnelle de figures sur support papier. Par conséquent, la phase d'entraînement, qui représente la durée nécessaire à l'élève pour maîtriser l'outil numérique, peut s'avérer longue et fastidieuse. Se pose aussi le problème de la *transférabilité* de l'apprentissage. Une connaissance acquise sur le logiciel pourrait ne pas s'avérer persistante lors du transfert vers le papier. **L'idée de notre travail est d'avoir les bénéfices du numérique tout en simulant l'approche papier-crayon pour assurer le fait que l'élève puisse naviguer entre les supports sans perdre ses acquis**. Pour cela, il est primordial que le support numérique soit le plus transparent possible à l'élève. Il faut donc permettre à l'utilisateur de construire librement ses figures géométriques, **en simulant le plus possible la condition traditionnelle papier-crayon**. Dans l'idée de fluidifier le processus de résolution d'exercices, et d'éviter une phase d'entraînement trop importante, l'utilisation de

la tablette orientée stylet est tout à fait appropriée. Nous profitons de l'interaction stylet-doigts offerte par les tablettes pour permettre le dessin libre avec le stylet, et de plus, proposer des outils virtuels réalistes que l'élève pourra manipuler avec ses doigts. Il en découle que si l'apprenant maîtrise un rapporteur virtuel sur l'application, il sera capable d'utiliser le rapporteur réel dans la condition traditionnelle. Pour pouvoir interpréter et évaluer la validité du dessin libre de l'élève par rapport à une consigne, il est primordial que notre système puisse **reconnaître, à la volée, les tracés manuscrits de l'élève et les transformer en éléments géométriques en temps-réel**.

1.3 L'intérêt de la supervision et du feedback immédiat

Le feedback consiste à fournir une observation sur le travail de l'élève, après évaluation, de manière à ce qu'il prenne conscience de ses acquis, mais aussi de ce qu'il peut améliorer, et ainsi être capable de modifier son approche pour la résolution d'un problème par exemple. Dans [Chi87], Chickering et Gamson démontrent l'importance du feedback comme bonne pratique pour l'éducation à l'école. Deux modes de génération de feedback se distinguent, le premier est le mode différé (*delayed*), et le second est dit immédiat (*prompt*). Le feedback différé est typiquement le retour de l'enseignant sur la copie à corriger d'un exercice de construction. Cela peut se traduire par un feedback global sur la forme finale de la figure. Cependant, l'enseignant n'a pas d'information sur la procédure de résolution suivie par l'élève pour arriver à sa figure finale. Par contraste, un feedback immédiat est donné à l'élève au fil de la réalisation de la figure. Ainsi, l'apprenant peut avoir des retours sur son protocole de construction, voir ce qu'il a bien réalisé mais aussi prendre conscience de ses erreurs tout au long de la résolution de l'exercice. Dans [KD96], Kluger et DeNisi illustrent l'impact de l'intervention du feedback immédiat dans la performance de l'apprenant. Les auteurs de [Chi87] confirment la pertinence de ce deuxième type de feedback en déclarant que "*l'évaluation sans feedback immédiat n'a pas un impact conséquent sur l'apprentissage*". De là vient l'intérêt du support numérique comme complément pour l'apprentissage de la géométrie : il est impossible physiquement pour un enseignant d'évaluer au fil de l'eau la production de tous les élèves et de fournir des feedback personnalisés et à la volée sur la procédure de résolution de l'exercice.

Il existe différents types de feedback : le feedback correctif qui se contente de décrire l'erreur de l'élève (contraintes non satisfaites de l'exercice par exemple), et le feedback

de guidage qui donne des pistes pour que l'élève avance dans la résolution (soit en corrigeant une erreur, soit en le guidant pour résoudre la prochaine étape). Dans [Bon+20], les auteurs démontrent l'impact positif de la génération de feedback de guidage pour l'apprentissage de l'écriture sur tablette orientée stylet. Il est cependant primordial que les feedback générés soient pédagogiquement solides et intelligibles pour l'élève, sous peine de ne pas produire d'effets positifs [PP16].

Pour mettre en avant ces grands principes pédagogiques, le système d' e-apprentissage doit avoir une connaissance du domaine de la géométrie pour pouvoir superviser les stratégies de résolution de l'élève. Pour ce faire, il est nécessaire que la reconnaissance à la volée des tracés manuscrits de l'élève soit couplée à une interprétation sémantique en temps-réel de ses actions relativement à l'instruction du problème. Pour l'apprentissage de la géométrie de construction, les feedback immédiats générés (par exemple une coloration en rouge si une partie de la figure est erronée) doivent amener l'élève à éviter la propagation des erreurs dans sa processus de résolution, de même que le guidage doit lui permettre de se sortir d'éventuelles impasses. Notre but est donc de concevoir un moteur de supervision capable d'évaluer et de guider les stratégies de résolution de l'élève. La modélisation de la connaissance de l'expert et celle de l'élève sont deux piliers des systèmes tutoriels intelligents, domaine de recherche à part entière en intelligence artificielle.

1.4 Bilan

Dans ce chapitre, nous avons présenté les fondements pédagogiques qui ont guidé notre projet. Pour atteindre ces enjeux, nous avons concu et développé le système "IntuiGeo" en des techniques de reconnaissances de formes et des techniques de tutorat. Cela nous a permis d'arriver à un système tutoriel intelligent orienté stylet intuitif qui offre un parcours autonome et personnalisé à l'élève grace à la mise en place de feedback de correction et de guidage. Dans les prochains chapitres de cette partie, l'étude de l'état de l'art de ces deux domaines est présentée, ce qui nous permettra de dégager les grandes lignes de notre approche.

Chapitre 2

Interprétation à la volée de tracés manuscrits

Dans ce chapitre, nous nous focalisons sur le premier axe de notre travail : la reconnaissance à la volée de figures géométriques dessinées à main levée sur tablette numérique. Nous nous plaçons donc dans le contexte de l'interprétation en ligne de documents manuscrits structurés. Le début de ce chapitre présente les concepts généraux relatifs à ce domaine. Nous nous focaliserons ensuite sur les systèmes de reconnaissance en-ligne, permettant une interprétation à la volée des tracés de l'utilisateur. Cette présentation donne une vue d'ensemble de la problématique et des méthodes d'interprétation en ligne de documents structurés, avec une orientation particulière sur celles qui sont la base de notre travail.

2.1 Définitions générales

2.1.1 Documents structurés

Ce type de document est caractérisé par une structure prédéfinie. Il est constitué d'un ensemble de symboles bi-dimensionnels organisés selon une logique structurelle et sémantique. Les relations spatiales entre les éléments d'un document structuré, basées sur des conventions pré-établies (propriétés des cercles circonscrits par exemple dans la figure 2.1), donnent une information importante permettant l'interprétation du document. Par exemple, un schéma de chimie moléculaire, un circuit électrique ou un plan architectural sont considérés comme des documents structurés. Par contraste, une oeuvre d'art surréaliste ne l'est pas si elle ne suit pas de règles identifiables (*c.f.* figure 2.2).

Un document structuré peut se présenter sous des formes diverses : il peut être imprimé (image), ou décrit sous une forme numérique dynamiquement interprétable (séquence de tracés). Dans la prochaine section, nous présentons les différentes manières de composer

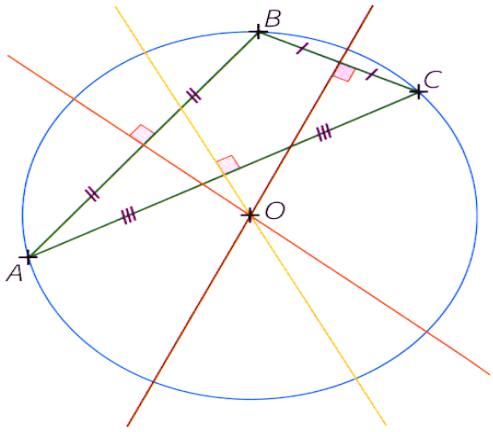


FIGURE 2.1 – Document structuré : un enseignant se base sur la propriété des cercles circonscrits pour dessiner cette figure géométrique

FIGURE 2.2 – Document non structuré : Joan Miro s'est inspiré de ses hallucinations dues à la faim pour créer son oeuvre : le carnaval d'Arlequin

de ce type de document.

2.1.2 Composition de documents structurés

La composition de document numérique se base sur deux modes d'interaction/de création, soit via la sélection et le glisser-déposer d'un ensemble de symboles prédéfinis, soit par la composition à main levée de l'ensemble des symboles. Composer un document structuré dans le premier mode consiste à utiliser des logiciels basés sur une interaction de type "WIMP" (Windows, Icons, Menus, Pointer). Un menu contenant des symboles est disponible dans l'interface et l'utilisateur va cliquer sur le symbole et le placer dans son schéma. La plupart des logiciels de géométrie dynamique présentés dans le chapitre précédent se basent sur ce principe. Comme évoqué précédemment, cette approche a des limites, notamment dans un cadre d'apprentissage où la composition est le cœur même d'un problème à résoudre pour un élève (manque de liberté dans la composition, et temps d'apprentissage du logiciel). Dans un tel contexte pédagogique, le deuxième mode est donc plus pertinent. Dans ce mode où l'utilisateur crée et compose ses propres symboles, nous distinguons deux façons de créer un document numérique.

La première façon de composer le document consiste à permettre le dessin libre de l'utilisateur sur papier, sans contraintes particulières. Le document est ensuite numérisé et interprété. La figure 2.3 illustre ce processus dit de *rétroconversion* [Gho12]. Le document

manipulé est alors représenté par un signal dit hors-ligne, et l'image est représentée par une matrice de pixels. L'interprétation consiste à retrouver les tracés dans le document, les isoler pour les reconnaître, et de les associer pour en déduire le sens de la structure du document (dans l'exemple de la figure 2.3, les pièces, les étages, etc). Cette méthode d'interprétation est dite *hors-ligne*. On distingue deux approches pour interpréter un document hors-ligne :

— Approche classique (en batch), le système interprète le document à partir d'une base de symboles pré-établie et un modèle statistique (*e.g.* dans [KAO14], ou [AS10]), où à partir de la modélisation a priori de la connaissance du domaine à partir d'un langage visuel (*e.g.* dans[MSL06] ou [RBE00]), sans interaction avec l'utilisateur, et lui affiche le résultat de l'analyse.

— Approche incrémentale, *i.e.* au fur et à mesure que le système reçoit en entrée de nouvelles données (par exemple à travers un flux de documents), il est capable d'apprendre de nouveaux concepts (classes), d'en oublier ou d'en fusionner d'autres [Ngo+17]. Certaines approches impliquent une interaction avec l'utilisateur. Quand le système détecte une ambiguïté entre deux symboles (par exemple entre un lit et un canapé), l'utilisateur peut être sollicité pour choisir la bonne interprétation afin d'éviter la propagation des erreurs de reconnaissance, et aussi permettre au système d'apprendre de nouveaux de symboles [MAH00] [Gho12] [WZY07].

a) Plan manuscrit

b) Plan numérisé et interprété

FIGURE 2.3 – Processus de rétro-conversion [Gho12]

La deuxième façon de composer un document structuré est celle que nous adopterons dans cette thèse, elle correspond à une composition interactive en ligne. Plus précisément, l'utilisateur utilise un stylet sur un écran tactile par exemple, le signal à interpréter est alors un signal dit *en-ligne*. Il contient des informations qui facilitent le processus

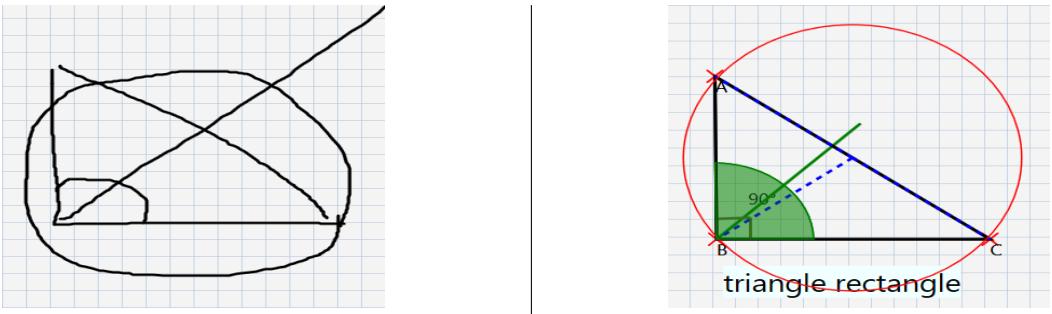
d'interprétation, notamment l'ordre des tracés manuscrits, les coordonnées des points entre posers et levers de stylos, ou encore la pression exercée sur l'écran de la tablette. Nous pouvons distinguer deux approches pour analyser un document en-ligne : une approche dite *a posteriori* et une approche dite *à la volée*. Nous nous y intéressons en détail dans la section suivante.

2.1.3 Interprétation en ligne de documents manuscrits

Chacune des approches d'interprétation en-ligne (a posteriori ou à la volée) engendre une interaction différente entre le système de reconnaissance et l'utilisateur.

Interprétation a posteriori

Dans le cas d'une interprétation a posteriori (on parle de *lazy interpretation*), le système interprète le schéma après sa réalisation complète, et à la demande de l'utilisateur. La figure 2.4 illustre cette approche dans le contexte de la composition de figures géométriques.



(a) (b)

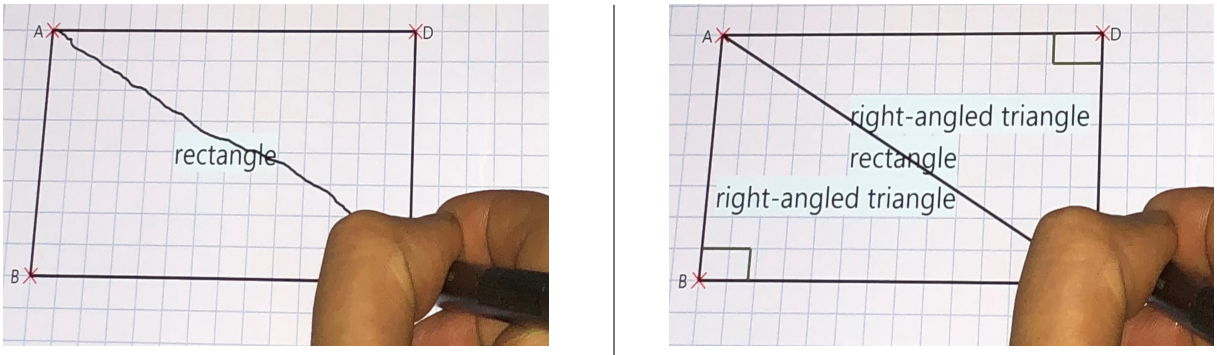
FIGURE 2.4 – Interprétation a posteriori : figure manuscrite (a), figure interprétée et remise au propre (b)

L'avantage de cette approche est que le système a une connaissance complète du document, notamment l'agencement spatial des tracés, appelé **contexte structurel**. Nous verrons par la suite que la connaissance du contexte structurel peut s'avérer précieuse pour arriver à une bonne interprétation du schéma de l'utilisateur. En revanche, le risque majeur est la propagation de l'erreur, puisque la mauvaise interprétation d'un tracé entraînera forcément d'autres erreurs. De plus, comme nous l'avons explicité auparavant, nous voulons pouvoir générer un feedback immédiat pour accompagner l'élève dans sa

réalisation. Pour ce faire, il faut que l'interprétation des tracés soit réalisée au fil de l'eau et en temps réel, il s'agit alors d'une interprétation à la volée.

Interprétation à la volée

Lors d'une interprétation à la volée (on parle de *eager interpretation*), les tracés sont interprétés au fil de l'eau, chaque tracé ajouté est analysé au fil de l'au. De plus, un retour visuel est fourni à l'utilisateur lui indiquant comment le symbole a été interprété, par exemple une remise au propre ou une "beautification" (lissage du symbole mais en gardant dans une certaine mesure la spécificité de l'écriture de l'utilisateur). La figure 2.5 illustre le processus de production de schéma de géométrie sur une tablette orientée stylet. L'avantage de cette technique est que l'utilisateur devient acteur du processus



a) Tracé dessiné

b) Tracé interprété et remis au propre

FIGURE 2.5 – Composition en ligne et à la volée de figure géométrique sur tablette

d'interprétation, puisqu'il peut valider implicitement l'interprétation du système ou bien la rejeter explicitement en l'effaçant par exemple. Cette validation ou ce rejet limite le problème de la propagation des erreurs. Cependant, cette validation-rejet peut être vue comme un inconvénient, particulièrement si l'utilisateur est sollicité un grand nombre de fois dans le processus d'interprétation, ce qui pourrait le gêner.

Discussion

Puisque nous nous plaçons dans un contexte pédagogique, la composition à main levée semble la plus pertinente, donnant à l'élève un certain degré de liberté dans son dessin. De plus, l'interprétation à la volée est plus intéressante pour la génération de feedback

immédiats (plus les feedback interviennent tôt mieux c'est). La section suivante présente donc différents travaux autour de l'interprétation en-ligne.

2.2 Approches pour l'interprétation en-ligne de diagrammes manuscrits

L'interprétation en ligne de formes manuscrites est plus communément appelée reconnaissance de tracés (*sketch recognition*). Le terme tracé est utilisé par opposition au terme image qui est le signal d'entrée pour les méthodes de reconnaissance hors-ligne. Nous pouvons distinguer deux grandes approches pour l'interprétation de diagrammes manuscrits, les approches *statistiques* et les approches *structurelles*.

Les approches statistiques se basent sur des méthodes d'apprentissage automatique capables déterminer une forme dans son ensemble. Le principe consiste à extraire un ensemble de caractéristiques pour décrire la forme, cet ensemble représentant la signature de la forme. Les approches classiques suivent un processus en deux étapes :

— Extraction des caractéristiques ;
— Entraînement d'un modèle statistique sur une base de données.

Le modèle ainsi construit sera capable de classifier les formes : symboles, écritures, croquis... La tâche la plus importante est donc l'apprentissage de caractéristiques discriminantes. Les méthodes classiques se basent sur extraction "empiriques" de caractéristiques. On peut distinguer deux types de caractéristiques :

— *Caractéristiques statiques* : ces caractéristiques s'intéressent à l'aspect visuel de la forme, et ne tiennent pas compte de la nature dynamique du tracé, ce qui les rend robuste à la variabilité des styles de composition des symboles (un même schéma peut être composé de manière différente selon le scripteur). Ces caractéristiques sont traditionnellement utilisées pour la reconnaissance hors-ligne d'images (telles que SIFT [Low04] ou HOG [DT05]) ;
— *Caractéristiques dynamiques* : ces caractéristiques prennent en compte la nature en-ligne du signal du tracé (*e.g.* premier et dernier point, proportion des tracés descendants, etc).

Dans [DA13], un ensemble de 49 descripteurs combinant ces deux types de caractéristiques a été couplé à un classifieur de type SVM (Support Vector Machine) [Wan05], avec comme résultat une méthode générique pour l'interprétation en-ligne de symboles manuscrits. Pour démontrer la générécité de leur méthode, les auteurs évaluent les performances

en terme de reconnaissance sur des bases de symboles de différentes natures (*c.f.* figure 2.6)). D'autres types de systèmes de classification ont été utilisés dans le même contexte,

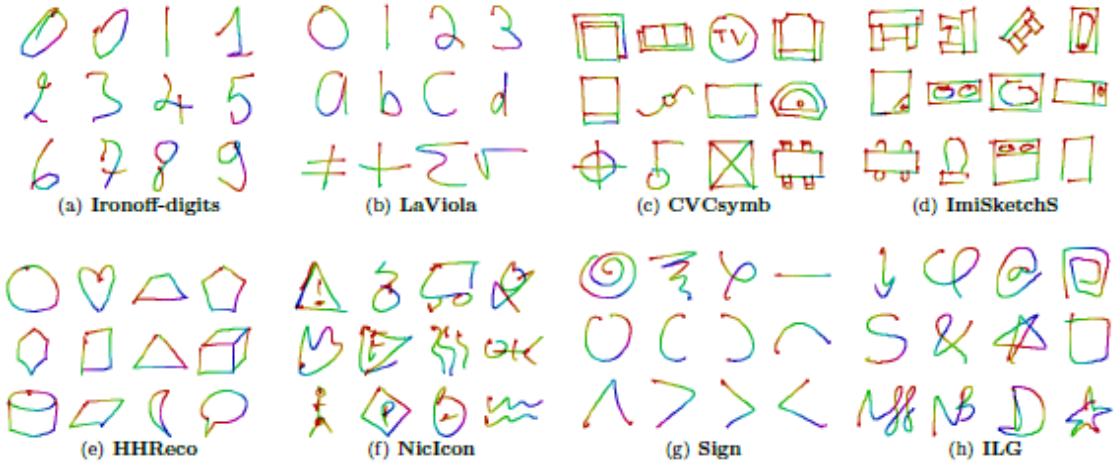


FIGURE 2.6 – Bases de symboles de tracés en-ligne utilisées dans [DA13]

tels que les réseaux de neurones, les K-NN (k plus proches voisins), les arbres de décision, ou encore les modèles de Markov cachées. Plus récemment, l'avènement des Réseaux de Neurones Profonds (Deep Neural Networks) a permis l'émergence de plusieurs méthodes pour la reconnaissance en ligne de tracés [Yu+15], [Zha+20], [WCZ18], [XJB19]. Plutôt que se baser sur un savoir faire empirique pour la définition des caractéristiques, les RNP sont capables de les extraire automatiquement à partir des données, et ont une très bonne performance si la base d'apprentissage est très grande.

Les approches statistiques sont les mieux adaptées pour la reconnaissance de formes *isolées* (*e.g.* reconnaissance en-ligne d'écriture manuscrite dans plusieurs langues dans [Car+20], ou reconnaissance en-ligne et à la volée de dessins de différentes natures dans [SKV16]). Cependant, elles ne permettent pas une modélisation structurelle explicite du document, qui est fondamentale dans notre contexte de composition de figures géométriques. En effet, pour pouvoir interpréter un schéma complexe, une modélisation explicite de la structure est nécessaire, pour pouvoir décomposer les formes, et analyser finement les relations spatiales et géométriques qui les lient. La nature de l'interprétation à la volée, tracé par tracé, et la nature fortement structurée des figures géométriques nous fait donc pencher vers la seconde famille d'approches structurelles, qui se basent sur une formulation des symboles en primitives graphiques (par exemple, un triangle est défini par trois primitives graphiques, des segments, caractérisées par leurs relations spatiales). C'est

pour cela que nous focalisons la suite de notre état de l'art sur la famille des méthodes structurelles qui est plus adaptée à notre contexte.

2.3 État de l'art des approches structurelles

Les approches structurelles pour l'interprétation de tracés manuscrits à main levée ont fait l'objet de beaucoup de travaux dans la littérature. En général, ces approches s'appuient sur deux principes : la formalisation de la connaissance a priori du domaine d'application à partir d'un formalisme grammatical, et l'analyse syntaxique qui interprète le signal en entrée (le tracé manuscrit dans notre cas).

2.3.1 Grammaires formelles

A l'origine, les grammaires formelles sont dédiées à l'interprétation des chaînes de caractères, pour, entre autres, modéliser les règles de composition d'un langage. On parle alors de *grammaire unidimensionnelle*.

Definition 2.3.1 (Grammaire formelle). Une grammaire formelle classique est définie en tant que tuple G=(V_N, V_T, S, P) avec :

- V_N : l'ensemble fini de symboles non terminaux ;
- V_T : l'ensemble fini des symboles terminaux ;
- S : le symbole initial, ou axiome ;
- P : l'ensemble des règles de productions,
  avec p $\in$ P la forme $\alpha \rightarrow \beta$, avec α, $\beta \in (V_T \cup V_N)^*$.

α est dénommé partie gauche de la règle, β partie droite.

Definition 2.3.2 (Dérivations et réductions). Une *dérivation* consiste à remplacer un élément de la partie gauche par un élément de la partie droite. Une *réduction* consiste à remplacer un élément de la partie droite par celui de la partie gauche. Dans le cadre de la reconnaissance de tracés, l'ensemble V_N représente les classes de symboles du domaine d'application. Par conséquent, c'est le processus de réduction que l'analyseur syntaxique associé à la grammaire utilise.

Definition 2.3.3 (Phrases et langages d'une grammaire). Une *phrase* est une combinaison d'éléments terminaux et non terminaux, sous la forme $phr \in (V_T \cup V_N)^*$. Elle peut être générée par une séquence de dérivations à partir du symbole de départ S. Une

phrase terminale est une phrase contenant exclusivement des éléments terminaux, sous la forme $phr_t \in V_T{}^*$. Le *langage* de la grammaire G, dénoté *L(G)*, est l'ensemble des phrases terminales, générées par la grammaire.

Le linguiste Noam Chomsky a distingué quatre classes de grammaires formelles [Cho56], nous en citons deux ici qui ont un intérêt pour notre propos.

Definition 2.3.4 (Grammaires hors-contexte). Ces grammaires ont des règles de production de la forme $\alpha \rightarrow \beta$ tels que $\alpha \in V_N$ et $\beta \in (V_T \cup V_N)^*$.

Definition 2.3.5 (Grammaires contextuelles). : ces grammaires ont des règles de production de la forme $x\alpha y \rightarrow x\beta y$, avec x et y $\in (V_T \cup V_N)^*$. Cela veut dire que la dérivation de α en β dépend de son contexte, défini par les éléments x et y.

Example 2.3.1. La grammaire hors-contexte G permet de générer des phrases en langue française. Elle est définie par :

- V_N : groupe-nominal, groupe-verbal, nom-propre, article, nom-composé, nom-commun, verbe ;
- V_T : Omar, manuscrit, mange, pomme, le, la ;
- S : le symbole initial ;
- P : l'ensemble des règles de productions :
  - S → <groupe-nominal> <groupe-verbal> ;
  - groupe-nominal → <nom-propre> | <nom-composé> ;
  - nom-composé → <nom-propre> | <article> <nom> ;
  - groupe-verbal → <verbe> <groupe-nominal> ;
  - nom-propre → Omar ;
  - nom-commun → manuscrit | pomme ;
  - verbe → mange | rédige ;
  - article → le | la.

Cette grammaire générera les phrases "*Omar rédige le manuscrit*", ou "*Omar mange la pomme*", mais aussi "*Omar mange le manuscrit*", une phrase correcte mais qui n'a pas de sens (même si, pour être honnête, il y a plus de chances que je mange mon manuscrit qu'une pomme).

En se basant sur la définition des grammaires classiques unidimensionnelles, plusieurs travaux se sont intéressés à l'adaptation de ces formalismes à la reconnaissance de tracés manuscrits. Le passage du contexte uni-dimensionnel au bidimensionnel fait que ces

formalismes sont appelés grammaires bidimensionnelles, ou encore grammaires visuelles. Ces grammaires visuelles peuvent être réparties en trois catégories détaillées ci-après : les grammaires à base d'opérateurs, les grammaires à base de fonctions, et les grammaires à base de graphes.

2.3.2 Grammaires à base d'opérateurs

Dans une grammaire à base d'opérateurs, une règle est de la forme c → a + b. Un exemple bien connu dans la littérature est la grammaire PDL (*Picture Description Langage*) [Sha70] caractérisée par quatre opérateurs (+, x, -, *), illustrés dans la figure 2.7. Ces opérateurs représentent une relation particulière entre les deux éléments qui les entourent. Pour la reconnaissance de tracés manuscrits, les symboles sont décrits par leurs extrémités (tête -t- et queue -q-). Ainsi les éléments entourant un opérateur sont des extrémités de segment. La concaténation de deux symboles est alors réalisée par réduction d'une règle, selon la relation représentée par l'opérateur utilisé. Bien que cette grammaire fasse partie des précurseurs des approches syntaxiques de reconnaissance de formes, elle a été très peu utilisée dans ce contexte.

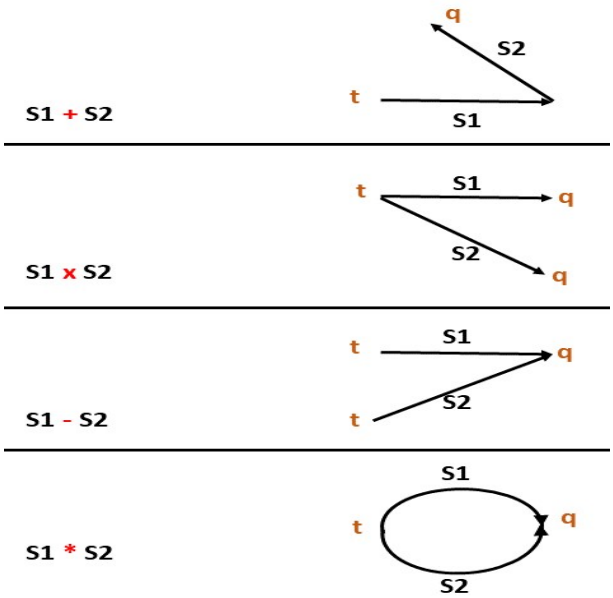


FIGURE 2.7 – Opérateurs de concaténation pour la grammaire PDL [Sha70]

Parmi les grammaires bidimensionnelles à base d'opérateurs, nous pouvons aussi trouver la définition du formalisme EPF (*Enhanced Positional Formalism*) [Coü05], qui se

base sur la définition d'un opérateur de position entre deux symboles et qui s'écrit sous la forme α ***at (Pos)*** β, ce qui signifie que β se trouve à la position *Pos* de α. De plus, un opérateur de factorisation * permet de modéliser des relations entre plusieurs symboles dans la partie droite de la règle. La figure 2.8 illustre la règle de production d'un triangle dans ce formalisme (nous pouvons remarquer que c'est bien la réduction qui est considérée ici. Pour obtenir le triangle T, il faut que les 3 segments S1, S2, S3 soient inter-connectés par leurs extrémités selon "un ordre" particulier).

Triangle T $\rightarrow$
S1 at (extrémité) S2 * S2 at (extrémité) S3 * S3 at (extrémité) S1

FIGURE 2.8 – Règle de production de triangle en EPF

L'analyseur syntaxique associé au formalisme EPF, dénommé DMOS, est un analyseur de type *descendant / top-down* basé sur la logique de premier ordre. L'analyse descendante consiste à partir de la connaissance modélisée dans les règles, pour prédire la présence de primitives graphiques et effectuer des segmentations contextuelles. Bien que ce formalisme soit généralement utilisé dans un contexte de reconnaissance hors-ligne, il a aussi été adapté dans un contexte en ligne pour l'interprétation de diagrammes [Lem+13].

D'autres formalismes, *e.g.* les grammaires de position (*Positional Grammars*) [Cos+93], se basent sur le même principe. Une règle de production dans ce formalisme est de la forme :

$$\alpha \rightarrow \beta_1 \ \mathrm{R}_1 \ \beta_2 \ \mathrm{R}_2 \ \beta_3 \ ... \ \mathrm{R}_{n-1} \ \beta_n.$$

tel que R_i est le positionnement de β_i par rapport à son antécédent β_{i-1} dans la production. Ce type de formalisme présente cependant une expressivité limitée puisqu'on peut exprimer le positionnement d'un élément uniquement par rapport à son antécédent. Cette limite a été éliminée dans l'extension de formalisme, nommée grammaires de position étendues (*eXtended Positional Grammars*) [CP00].

Les grammaires de position étendues ont été adaptées pour l'interprétation de croquis manuscrits à la volée, tels que les diagrammes UML, dans [Cos+04]. Les terminaux de cette grammaire sont alors les primitives graphiques (segment, arc, cercle, etc). L'analyseur syntaxique essaie donc de décomposer, ou segmenter, les tracés en entrée en primitives graphiques, pour ensuite reconnaître les formes composites (*e.g.* les classes en UML). Ce type d'analyse est dit *ascendant / bottom-up* : l'arbre d'analyse est construit à partir

des feuilles qui représentent les symboles terminaux V_T, pour trouver des symboles non terminaux V_N dans les différents niveaux jusqu'à arriver à la racine. Les grammaires de dessin (Sketch Grammars) [CDR05] étendent les grammaires de position en introduisant la notion de relation temporelle en plus des relations spatiales. Elles ont été exploitées dans [DR20] pour l'interprétation en lignes de diagrammes manuscrits.

Dans [ÁSB14], les auteurs proposent une extension de la grammaire unidimensionnelle SCFG (Stochastic Context Free Grammar) par la définition de 2D-SCFG pour l'interprétation des expressions mathématiques. SCFG est une grammaire hors-contexte augmentée d'une fonction de probabilité d'application pour chaque règle. Les règles de production dans cette grammaire sont de la forme $A \xrightarrow{r} a$, tel que r représente une relation spatiale (*e.g.* horizontal, vertical, dans, etc). Cette méthode propose une combinaison entre approche statistique et structurelle. En effet, sont utilisés les réseaux de markov cachés pour la reconnaissance des symboles mathématiques et les SVM (Séparateur à vastes marges) pour la classification des relations spatiales dans la grammaire. La méthode d'analyse est basée sur l'algorithme CYK [You67], analyseur ascendant dédié aux grammaires hors-contexte classiques. Dérivant directement des SCFG hors-contexte, le nombre maximum de paramètres dans la partie droite d'une production est limité à deux, ce qui réduit l'expressivité du formalisme.

Ces grammaires à base d'opérateurs se caractérisent donc par la simplicité de la formulation des règles, ce qui facilite pour l'expert non développeur la modélisation de la connaissance a priori du domaine d'application. Cependant, cette simplicité de la formulation a pour revers le manque d'expressivité. Cette catégorie de formalismes à base d'opérateurs ne permet pas de moduler la rigidité des relations spatiales, cette modulation est pourtant nécessaire à cause de la variabilité et l'imprécision des dessins réalisés à main levée des utilisateurs.

2.3.3 Grammaires à base de fonctions

Cette catégorie de grammaires se base sur des fonctions, en lieu et place des opérateurs, pour modéliser les relations entre les symboles. Dans ce type de grammaires, les règles de productions sont de la forme $\alpha \rightarrow$ *{β, γ} fonction(β, γ)*. Dans cettte catégorie, nous trouvons par exemple les grammaires d'adjacence [JG95] dont les règles de production sont de la forme :

$$\alpha \rightarrow \{\beta_1, ..., \beta_j\}, \text{ si } c_1,...c_k,$$

avec $\beta_j \in (\mathrm{V}_T \cup \mathrm{V}_N)$, $\alpha \in \mathrm{V}_N$, et c_k un ensemble de contraintes définies sur l'ensemble des objets présents dans la partie droite de la règle. Les symboles de ce formalisme sont définis par un ensemble d'attributs $\mathrm{A} = \mathrm{a}_1, ..., \mathrm{a}_n$, modélisant leurs caractéristiques (les coordonnées des extrémités par exemple pour le cas d'un segment). Ces grammaires sont dites d'adjacence parce-que les fonctions représentent la proximité graphique des symboles. Cela permet à l'analyseur syntaxique associé au formalisme de ne regrouper que les éléments considérés comme proches pour limiter les calculs et contrôler la combinatoire. Une adaptation de cette grammaire à l'interprétation à la volée a été proposée dans [Mas+10]. Le processus d'analyse y est similaire à celui des grammaires de positions étendues au sens où il est ascendant. Chaque nouveau tracé dessiné par l'utilisateur est interprété. Une approximation polygonale est d'abord effectuée pour segmenter le tracé en primitives graphiques et les donner en entrée à l'analyseur. À partir d'une primitive graphique, une recherche de toutes les productions qu'elle peut réduire, *i.e.* les règles contenant ce type de primitives dans leur partie droite, est effectuée.

Dans [Mar94], les auteurs définissent une nouvelle classe de langages visuels, dénommée grammaires de multi-ensembles à contraintes (GMC). Les règles de production y sont de la forme :

$$\alpha \rightarrow \beta \left\{\text{Contraintes}\right\} \mid \alpha \in \mathrm{V}_N{}^{+},\ \beta \in (\mathrm{V}_T \cup \mathrm{V}_N)^{*}.$$

avec α et β des muti-ensembles d'éléments. L'originalité de l'approche réside dans le fait que les éléments de la partie droite peuvent créer plusieurs nouveaux symboles dans la partie gauche (lors d'une réduction). Ceci apporte plus d'expressivité au formalisme et est particulièrement bien adapté au domaine de la composition de figures géométriques (deux segments qui s'intersectent peuvent créer quatre nouveaux segments dans le document par exemple). Le processus d'analyse est aussi ascendant et similaire à celui des grammaires d'adjacence.

La limite de ce processus purement ascendant provient du fait que l'analyseur doit tester toutes les combinaisons possibles d'éléments de la partie droite des productions pour ensuite évaluer la validité des contraintes et trouver les réductions valides. Cela implique qu'au fur et à mesure que le document se complexifie, le temps d'analyse s'accroît jusqu'à ne plus respecter la contrainte d'interprétation temps-réel.

Pour contourner cette limite, une extension des GMC a été proposée dans [MA09] ; elles sont dénommées : Les Grammaires Multi-ensembles à Contraintes Pilotées par le Contexte (GMC-PC). Nous présentons ici succinctement les principes de base du formalisme GMC-PC. Nous les développerons plus en détail dans le chapitre 4. Une règle de production

dans ce formalisme est définie comme suit :

$$\alpha \rightarrow \beta \begin{Bmatrix} \text{Préconditions} \\ \text{Contraintes} \\ \text{Postconditions} \end{Bmatrix} \mid \alpha \in \mathrm{V}_N{}^+,\ \beta \in (\mathrm{V}_T \cup \mathrm{V}_N)^+.$$

Une particularité de cette grammaire est que l'alphabet est limité au tracé manuscrit de l'utilisateur, au contraire de la plupart des approches syntaxiques où l'alphabet est composé de primitives graphiques, *i.e.* $\mathrm{V}_T = \{\text{Tracé}\}$. Cela permet d'éviter la segmentation systématique des tracés en primitives graphiques, qui peut être source d'erreurs. En effet, si la segmentation systématique des tracés est pertinente pour la reconnaissance de symboles fortement structurés, elle ne l'est pas pour les symboles complexes peu structurés. (*c.f.* figure 2.9). Un des avantages de cette approche est la possibilité de reconnaître directement des formes complexes (*e.g.* avec un classifieur) sans forcément passer par une étape de segmentation. Dans les règles de production, les blocs de préconditions et post-

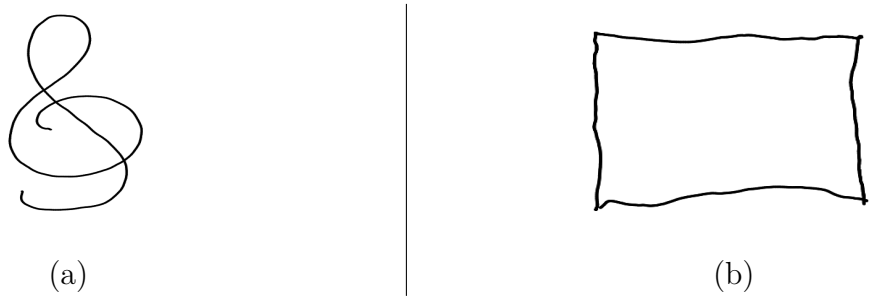


FIGURE 2.9 – Symbole complexe peu structuré : clé de sol (a), Symbole structuré : rectangle (b) [MA09]

conditions modélisent une vision globale du document ainsi que le contexte en se basant sur la position relative d'une partie d'un symbole (*e.g* extrémité d'un tracé) par rapport à une zone de contact d'un symbole de référence (logique similaire à celle des opérateurs de positions du formalisme EPF). Le bloc de contraintes quant à lui modélise une vision locale du tracé analysé. Ce bloc de contraintes peut contenir des contraintes statistiques et des contraintes structurelles. Plus précisément, une contrainte statistique peut faire appel à un reconnaisseur de formes pour interpréter le tracé (c'est le cas pour la clé de sol dans la figure 2.9.a). Des contraintes structurelles d'autres part peuvent être évaluées pour vérifier l'agencement structurel des constituants d'un rectangle par exemple (figure 2.9.b). Cette modélisation en blocs de la règle permet de guider l'analyse par le contexte

et de ne déclencher que les règles contextuellement valides. Le processus d'analyse est en fait une combinaison entre approche ascendante guidée par les éléments, et approche descendante guidée par les contextes (ou zones de contact dans le document). Deux stratégies d'exploration sont possibles : profondeur d'abord et largeur d'abord. La stratégie largeur d'abord permet de mettre en compétitions plusieurs interprétations possibles pour choisir la meilleure (le résultat d'une interprétation d'un tracé est une séquence de règles de production). Par ailleurs, la stratégie profondeur d'abord permet de réduire la première règle applicable sans la mettre en compétition avec d'autres productions, ce qui permet d'alléger la combinatoire du processus d'analyse. Ce formalisme a été adapté avec succès à plusieurs domaines d'application, dont la composition de plans d'architecture [Gho12], [PMA10], la composition de circuits électriques, ou encore la reconnaissance de partitions musicales [MA09].

LADDER [HD05] est une autre approche générique pour la reconnaissance en ligne de tracés basée sur un langage de description permettant de modéliser la connaissance a priori du domaine. Comme la plupart des approches étudiées, le processus d'analyse se base sur la segmentation et le regroupement de primitives graphiques avec une analyse de type descendant. Ce langage de description peut être considéré comme un grammaire à base de fonctions : la connaissance du domaine est définie par des productions modélisant le fait que des éléments peuvent etre remplacées par d'autres si certaines contraintes (*e.g.* géométriques) sont satisfaites. Un framework générique d'analyse de tracés manuscrits, PaleoSketch [PH08], a été développé à partir de ce formalisme et adapté à plusieurs domaines d'application, tel que la composition de partitions musicales [TBH15], ou la construction de schémas pour les cours de mécanique [Ati+14].

2.3.4 Grammaires à base de graphes

Une grammaire de graphe est définie de façon similaire aux grammaires que nous avons étudié jusqu'ici, à ceci près que les primitives graphiques sont représentées par des noeuds, et les relations structurelles par des arcs. La figure 2.10 illustre deux exemples de règles de production dans une grammaire de graphes, définissant un triangle et un quadrilatère. La réduction d'une production dans le contexte de l'interprétation de tracés consiste donc à remplacer le graphe de la partie droite par celui de la partie gauche de la règle. Dans le cadre de l'interprétation de tracés, les graphes (d'adjacence) ont souvent été utilisés. C'est le cas par exemple, pour l'interprétation d'expressions mathématiques manuscrites dans [ZMV13], ou [ÁSB16]. D'autres sont plutôt axées sur l'interprétation

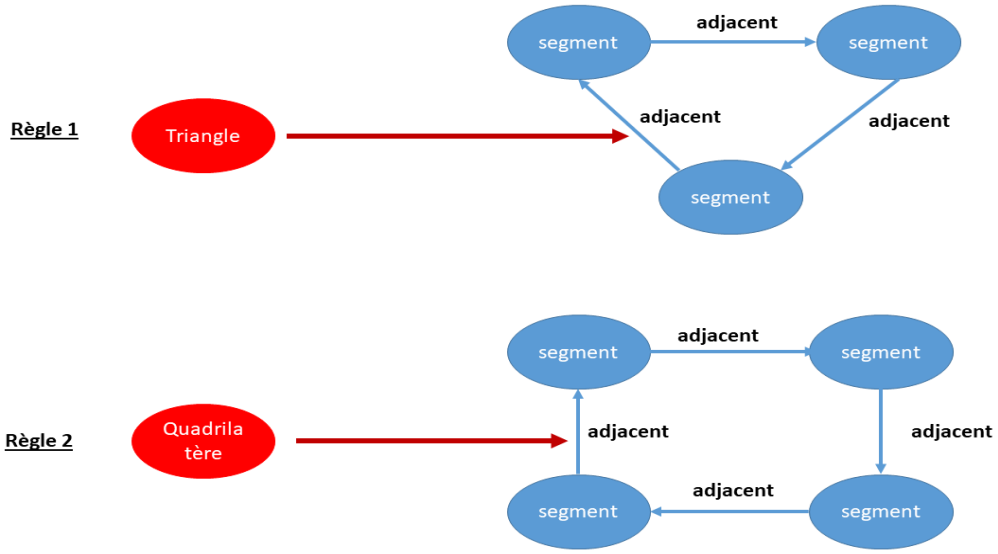


FIGURE 2.10 – Exemples de règles de production d'une grammaire de graphes pour la géométrie

de diagrammes en ligne [Che+15]. Dans [Jul+17], une méthode générique basée sur une grammaire de graphes est proposée et appliquée aussi bien à l'interprétation en ligne de diagrammes qu'à l'analyse des expressions mathématiques. L'approche est toutefois a posteriori et non pas à la volée. Les auteurs proposent une méthode d'analyse d'un croquis sur plusieurs étapes : premièrement la segmentation et le regroupement des tracés pouvant constituer des symboles, et la reconnaissance des symboles par un classifieur. Ensuite les relations entre les symboles sont identifiées par un classifieur de relations et un graphe d'hypothèses est généré. Le processus d'analyse consiste à construire des arbres représentant les interprétations possibles en appliquant les règles de la grammaire modélisant la connaissance a priori. Les méthodes basées sur les graphes ont été appliquées avec succès sur d'autres domaines d'application, tel que la reconnaissance des plans d'architecture [HTL15], ou l'interprétation des partitions musicales [FB93].

2.3.5 Bilan

Comme nous l'avions mentionné, les approches statistiques pures ne sont pas adaptée à l'interprétation à la volée de schémas géométriques puisqu'elles ne permettent pas de modéliser de manière explicite la structure du document. Les approches structurelles semblent par contre bien adaptées à notre besoin de modéliser la connaissance du domaine

et la structure du document. En effet, les figures géométriques sont fortement structurées et chaque élément peut être défini par ses constituants et par les relations spatiales qui les régissent. Les formalismes à base de graphes se caractérisent par une grande expressivité. Cependant, ils sont difficiles à manipuler pour l'expert non développeur, surtout si le domaine d'application est complexe. Dans le cadre du projet ACTIF, nous souhaitons que le concepteur (enseignant par exemple) soit, à terme, capable de modéliser la connaissance du domaine de façon intuitive, la complexité de manipulation des grammaires à base de graphe en est une limite claire. C'est pour cela que nous nous baserons sur le formalisme grammatical des GMC-PC, celui-ci offrant l'avantage de coupler approche structurelle pour modéliser globalement le document et approche statistique pour reconnaître localement la forme d'un tracé. La grammaire et son analyseur associé sont la base du framework DALI [MA09]. C'est un formalisme générique pour l'interprétation à la volée de tracés manuscrits. Dans le contexte de notre étude portant sur l'interprétation de figures géométriques tracées à main levée, nous devrons donc définir les règles de production relatives à la connaissance de la géométrie. Par ailleurs, comme nous le verrons plus en détails dans le chapitre 4, cette définition ne sera pas suffisante, il faudra aussi contrôler la complexité de la combinatoire du processus d'analyse, pour rester sur des temps d'interprétation rapides nécessaires à une interaction en temps-réel avec l'utilisateur (pour la génération de feedback visuels et correctifs immédiats). En effet, notre domaine d'application est caractérisé par une forte composante sémantique (la nature des figures, les différentes interactions entre éléments), ce qui rajoute de la complexité au niveau de l'interprétation qu'il faudra circonscrire au temps-réel. Le premier axe de ces travaux de thèse portera donc sur l'adaptation et l'extension de ce formalisme pour respecter la contrainte d'analyse à la volée, pour garantir l'interaction en temps-réel avec l'utilisateur, tout en conservant l'expressivité et la généricité de la grammaire.

Au-delà de l'interprétation à la volée en temps-réel des tracés, le second défi de cette thèse est la supervision de la résolution d'exercices de géométrie. Nous allons donc dresser, dans le chapitre suivant, un état de l'art des systèmes tutoriels intelligents pour positionner et justifier les approches que nous avons mises en oeuvre dans ce travail de thèse.

Chapitre 3

État de l'art sur les systèmes tutoriels intelligents

Dans ce chapitre, nous présentons les concepts relatifs au deuxième axe de cette thèse : la supervision en temps-réel des stratégies de résolution dans le contexte d'un exercice de construction en géométrie. Nous commencerons par les concepts généraux relatifs au domaine des systèmes tutoriels intelligents. En particulier, nous étudierons l'architecture globale de ces types de systèmes, celle-ci permet d'identifier les fonctionnalités attendues d'un tuteur. Nous nous intéresserons ensuite aux différentes approches de modélisation de la connaissance experte et de supervision du travail de l'élève, ce qui permettra de positionner notre approche en mettant en avant nos choix et nos contributions.

3.1 Définitions générales

3.1.1 Qu'est ce qu'un système tutoriel intelligent ?

Dans [Nas16], l'auteur définit un système tutoriel intelligent (ou STI) comme un logiciel qui a pour but de fournir une instruction ou feedback immédiats et personnalisés aux apprenants, sans intervention de l'enseignant. Un STI a pour objectif de faciliter l'apprentissage d'une manière efficace en utilisant la diversité des technologies de l'information. Pour les auteurs de [FO16], un STI doit fournir des activités d'apprentissage adaptées aux connaissances et acquis de l'élève dans le but de stimuler le processus d'apprentissage, en fournissant un feedback individualisé et en évitant la frustration et le désengagement de l'apprenant qui pourraient être induits par de mauvaises performances.

3.1.2 Historique des STI

Les ancêtres des STI, si on regarde l'évolution du domaine, sont les CAI (Computer Assisted Instruction) qui sont apparus dans les années 50, et étaient à la mode dans les années 60, parallèlement aux Beatles et aux cheveux longs. Les CAI ont vu le jour avec le principe de programmation linéaire : l'instruction est programmée sous forme d'étapes servant à guider l'apprenant vers un but précis. Ce type d'approche implique l'unicité de l'instruction et des étapes, et cela quelque soit la réponse de l'élève (elle était en fait ignorée). L'évolution en programmes avec branchements a paré à cette limitation et permis une adaptation des instruction aux réponses de l'élève (ainsi un arbre d'apprentissage unique est généré pour chaque utilisateur). À la fin des années 60, les CAI génératifs ont vu le jour : l'idée est que le programme tutoriel doit être capable de créer et de résoudre des exercices, par exemple dans domaines comme l'arithmétique [Uhr69]. Par la suite, Carbonell [Car70] a été le premier à proposer l'inclusion des techniques d'intelligence artificielle dans le domaine des CAI, et c'est cette inclusion qui a donnée naissance aux STI. En effet, elle permis de répondre aux problèmes fondamentaux d'un environnement d'apprentissage numérique : chaque système doit avoir une représentation de *ce qui est enseigné, à qui est destiné l'enseignement, et comment enseigner* [Sel74]. C'est dans cet esprit que le domaine se développe dès lors avec l'émergence de la représentation des connaissances, la modélisation de l'apprenant, le tutorat socratique (instauration de dialogue entre le système et l'apprenant à des fins pédagogiques), ou encore les graphes génétiques [MA19]. Ce domaine évoluant sur une plage de temps assez longue et faisant intervenir plusieurs disciplines telles que la psychologie cognitive et la didactique, notre étude de l'état de l'art ne saurait être exhaustive. Nous nous concentrerons sur les fondements d'un STI, notamment l' architecture la plus commune, puis nous nous focaliserons sur des approches liées à notre travail.

3.1.3 L'architecture en quatre composants des STI

Dans [NMB10], les auteurs indiquent qu'une prolifération des STI a pu être observée dans la période 1990-2010. Bien que les architectures varient grandement entre les systèmes, une architecture générique (présentée sur la figure 3.1) se dégage et est basée sur quatre composants communs : le module du domaine, le module de l'apprenant, le module de tutorat, et l'interface. Détaillons un peu chacun de ces composants.

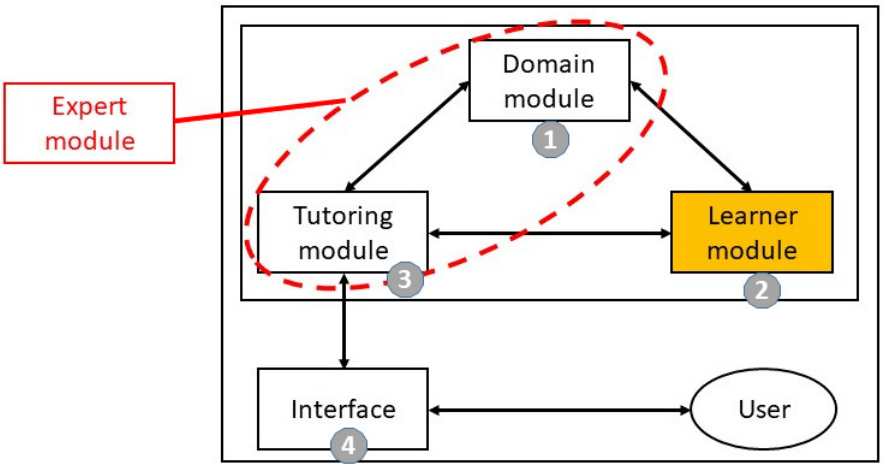


FIGURE 3.1 – L'architecture en quatre composants : 1) le module du domaine, 2) le module de l'apprenant, 3) le module de tutorat, 4) l'interface. [NMB10]

Le module du domaine

Il représente la connaissance du domaine d'application. Il peut prendre plusieurs formes, par exemple des règles, des réseaux sémantiques ou des contraintes.

Le module de l'apprenant

Il observe les actions de l'élève et a une représentation qualitative de l'état de résolution du problème. D'autres connaissances peuvent être représentées comme la modélisation de la mauvaise compréhension, de l'état affectif, ou de l'ensemble des acquis de l'élève dans un domaine particulier. L'objectif du module de l'élève est d'assembler le plus possible d'informations concernant l'élève. C'est cela qui permet d'ajuster le feedback en communiquant ces informations aux autres modules du STI. Deux approches se distinguent concernant le module de l'apprenant. D'un coté, des modules de l'apprenant dits à court-terme, qui se contentent de représenter l'état de résolution de l'élève sur un problème donné. De l'autre, les modules de l'apprenant à long terme, qui ont une représentation fine des connaissances de l'élève à partir de ses multiples utilisations du tuteur, ce qui permet par exemple de proposer automatiquement des exercices adaptés au niveau de l'élève.

Le module de tutorat

Parfois appelé module pédagogique, ce module a la connaissance nécessaire pour planifier des actions en conformité avec l'état du module de l'apprenant. Il est responsable de la stratégie d'enseignement, par exemple la génération de feedback, la génération de pistes (hints), ou l'établissement d'un dialogue de type socratique avec l'apprenant. Il est donc important que ce module soit pédagogiquement cohérent pour s'assurer des gains en terme d'apprentissage de l'élève. Certains travaux de la littérature fusionnent ce module avec celui du domaine pour créer un module expert complet [Py01] (*c.f.* élément rouge de la figure 3.1).

L'interface

Appelée aussi module de communication, elle représente l'environnement avec lequel interagit l'élève. Elle peut prendre plusieurs formes : interface graphique (dans le cas de la géométrie une zone de dessin avec boutons et menus), environnement de réalité virtuelle ou augmentée, ou langage naturel [TA16]. L'important est que ce module ne soit pas un obstacle à l'apprentissage. C'est pour cela que nous avons pris le pari de simuler l'approche traditionnelle papier-crayon avec l'utilisation de tablettes orientées stylet. Ce type de tablette propose un écran qui s'utilise comme le papier lorsque le stylet en est approché, et le stylet se tient comme un stylo, tout cela doit rendre le support numérique transparent à l'élève.

Après avoir présenté les concepts de base relatifs aux systèmes tutoriels intelligents, nous nous intéressons aux différentes approches de la littérature relatives à la représentation de la connaissance du domaine d'apprentissage (module du domaine), à l'évaluation des actions des élèves (module de l'apprenant), et aux différentes stratégies d'aides proposées (module pédagogique).

3.2 Approches et philosophies des STI

Dans le domaine des STI, deux approches prédominent : *les tuteurs à base de règles*, et *les tuteurs à base de contraintes*. Les tuteurs à base de règles modélisent la connaissance qui permet de générer une solution pas à pas, donc de trouver un plan de résolution. Les tuteurs à bases de contraintes, quant à eux, modélisent les conditions et les prédicats que chaque solution doit satisfaire. Ainsi, il existe deux philosophies différentes de l'apprentis-

sage : une qui cherche à simuler le processus de résolution de l'élève (quelles connaissances a-t-il appliquées pour arriver à une solution), et l'autre qui s'intéresse plutôt à l'état de résolution de l'apprenant (est-ce que sa production satisfait ou viole les contraintes de l'exercice).

3.2.1 Tuteurs à base de règles

Ces tuteurs ont comme fondement la théorie cognitive ACT (Adaptative Control of Thought) [And83]. Le tuteur intelligent doit en fait simuler le procédé employé par l'élève pour arriver à une solution. C'est pour cela que ce type de systèmes est considéré comme *centré sur le processus*. Ils produisent une description de la connaissance experte (les stratégies et les principes de résolution de problèmes).

Modélisation de la connaissance experte

La connaissance du domaine est représentée par des règles, dites expertes, qui modélisent la connaissance "idéale" (les théorèmes de géométrie par exemple) et les étapes que suivrait un expert pour arriver à une solution. On peut distinguer deux types de règles expertes :

— des règles de planification, qui modélisent la connaissance procédurale ;

— des règles d'opérateurs, qui modélisent la connaissance déclarative.

La connaissance procédurale représente le savoir-faire (comment résoudre un problème), tandis que la connaissance déclarative représente le savoir (la connaissance du domaine, et l'état du problème). La connaissance déclarative peut être représentée sous d'autres formes, par exemple les graphes conceptuels [BD04], les graphes de connaissance [Yan+18], ou cartes de concepts [Kum06]. Elle peut aussi être modélisée par des règles d'opérateurs. La figure 3.2 présente un exemple de règle experte de planification adaptée au domaine de la géométrie.

Après avoir présenté les principes de ces tuteurs à base de règle, nous nous intéressons maintenant aux différentes approches se basant sur ce paradigme.

Si le but est de dessiner [AB] perpendiculaire à [AC] **Alors**
Placer l'équerre sur [AB] et tracer le segment [AC]

FIGURE 3.2 – Exemple de règle experte

Approches basées sur le traçage de modèle et la modélisation explicite des erreurs

A partir de ce concept de modélisation de la connaissance, beaucoup de tuteurs se basent sur la technique de traçage de modèle (*model tracing*) [Ale10], très présente dans la littérature. Ces tuteurs sont communément nommés MTT (Model Tracing Tutors). Cette approche se base sur la combinaison des règles expertes explicitées plus tôt avec des règles erronées (*buggy rules*) qui modélisent les erreurs de compréhension (*misconceptions*) des apprenants. Cette combinaison a pour objectif de modéliser le plus explicitement possible le processus de résolution de l'élève. Un exemple de règle faussée est illustré dans la figure 3.3. Cette modélisation cognitive en règles expertes et erronées est associée à un

Si le but est de dessiner [AB] perpendiculaire à [AC] **Alors**
Placer le rapporteur sur [AB] et tracer l'angle AB,AC !=90°

FIGURE 3.3 – Exemple de règle erronée (buggy rule)

algorithme de traçage de modèle. Le traceur de modèle essaie de retracer le processus de résolution de l'élève en trouvant une séquence d'exécution des règles expertes et erronées qui correspondent à l'état de résolution du problème par l'élève. Si cette séquence est trouvée, on estime que le tuteur a compris le cheminement et le raisonnement de l'élève et a reconnu son plan. Le feedback généré correspond alors aux remplacements des règles erronées (*i.e.* suivies par l'élève) trouvées par les règles expertes (*i.e.* celles qu'il aurait du suivre). Nous étudions les systèmes qui se basent sur ce paradigme et dont les domaines sont proches du notre, *i.e.* les tuteurs qui gravitent autour de l'apprentissage des sciences en général.

Dans la littérature, l'un des systèmes parmi les plus aboutis est Cognitive Tutor [CMS00]. Ce tuteur se base sur l'approche de traçage de modèle explicitée plus haut. Il a été créé de prime abord pour le domaine de l'algèbre et la géométrie et a été adapté ensuite à plusieurs autres domaines tels que la génétique [Cor+10]. Cette généricité a contribué à créer un écosystème de STI basés sur le même paradigme de modélisation, appelés tuteurs cognitifs (Cognitive Tutors), et qui sont utilisés par plus de 500 000 étudiants annuellement aux USA [Ale10]. Au delà du traçage de modèle qui permet de suivre le chemin de résolution de l'élève, les tuteurs cognitifs utilisent la technique de *traçage de connaissance* [CMS00], basée sur un algorithme bayésien qui fournit une estimation des

acquis de l'élève. Cette technique s'explique simplement : une règle de production est soit apprise, soit non-apprise. Lorsqu'un élève applique une règle lors de la résolution d'un problème, elle passe à l'état apprise. L'aspect probabiliste vient du fait qu'il est possible que l'élève trouve accidentellement la solution sans que la règle soit apprise, et qu'il est aussi possible de faire une erreur, même quand la connaissance est acquise a priori. Cette modélisation ne prend cependant pas en compte l'aspect de l'oubli d'une compétence : une règle apprise ne peut plus passer à l'état non-apprise. La combinaison des deux techniques (traçage de modèle, traçage de connaissance) permet d'avoir une estimation la plus précise possible du comportement de l'apprenant et de fournir des feedback adaptés.

Andes [CGV02] est un tuteur intelligent pour l'apprentissage de la mécanique newtonienne (forces statiques, moments angulaires, etc), basé lui aussi sur les techniques de traçage de modèle et de traçage de la connaissance. Andes permet de dessiner des diagrammes, ou d'entrer des équations. Pour chaque problème, un solveur utilise une base de 540 règles expertes et erronées pour trouver les solutions possibles et générer un *graphe solution*, cette génération étant réalisée hors-ligne. Quand l'élève interagit avec l'interface, le tuteur essaie de faire correspondre ses actions avec un des noeuds du graphe, pour ensuite générer des feedback correctifs (flag feedback) ou des feedback de guidage sur la prochaine étape. Andes utilise aussi la technique de traçage de connaissance avec un modèle de l'apprenant modélisé par un réseau bayésien. Andes est intéressant car il permet de mettre en évidence quelques limites de ce type de tuteurs, au vu de son évolution à travers le temps. En effet, le traçage de connaissance, qui permet d'appréhender les compétences de l'élève sur le long terme, a été abandonné par les auteurs qui se sont rendus compte que cet aspect du système n'était pas en adéquation avec les besoins des enseignants [Col17]. De même pour la génération du graphe solution qui était trop lente et qui été remplacée par un solveur algébrique qui remplace les variables de l'équation avec les entrées de l'étudiant pour évaluer la validité des réponses et générer des feedback.

La limite principale des tuteurs à base de règles vient du fait qu'il est difficile de modéliser toutes les erreurs possibles de l'élève dans les règles erronées, surtout quand l'espace de connaissances incorrectes est vaste, ce qui est le cas pour la géométrie de construction. Les tuteurs à bases de contraintes adoptent une philosophie différente, qui justement ne représente pas l'espace des erreurs.

3.2.2 Tuteurs à base de contraintes

L'approche CBM (pour Constraints Based Modeling) a été proposée en premier lieu dans [Ohl94] et permet de pallier aux limites de l'approche à base de règles (plus spécialement le traçage de modèle et le traçage de connaissance). Au lieu de représenter l'espace des solutions correctes et incorrectes, ces tuteurs prennent le parti de représenter les contraintes qui doivent être satisfaites par toutes les solutions possibles. Chaque solution violant une de ces contraintes est incorrecte. Par conséquent, ces tuteurs ignorent la séquence d'actions suivie par l'élève et se focalisent sur l'état de la solution. A partir des contraintes du domaine non satisfaites, un feedback de correction peut donc être généré [Mit10]. Cette approche s'explique pédagogiquement par la théorie de Ollson sur l'apprentissage à partir des erreurs [Ohl96]. Cette théorie estime que la détection des erreurs de connaissance déclarative (modélisées par les contraintes non satisfaites) permet à l'élève de corriger sa méconnaissance procédurale (ses actions de résolution). Il en découle une différence notable avec les tuteurs à base de règles : **les CBT (Constraints Based Tutors) ne disposent pas de module expert**.

Modélisation de la connaissance par contraintes

Une contrainte est définie par un couple ($C_{pertinence}$, $C_{satisfaction}$), et est modélisée par la phrase : **Si** $C_{pertinence}$ est vraie, **alors** $C_{satisfaction}$ doit être vraie. Dans le cas contraire, la contrainte est violée et l'erreur est détectée. La figure 3.4 illustre un exemple de ce type de contrainte dans le domaine de la géométrie. Dans cet exemple, $C_{pertinence}$ = ABC est un triangle rectangle en A, et $C_{satisfaction}$ = (AB) $\perp$ (AC). La différence entre cette formalisation et celle des règles expertes dans les tuteurs cognitifs est qu'elle est évaluative, alors que les règles expertes sont génératives.

Si ABC est un triangle rectangle en A
Alors
(AB) $\perp$ (AC) ($\mathbf{C}_{satisfaction}$)

FIGURE 3.4 – Exemple de modélisation à base de contraintes

Ces contraintes modélisent en même temps les connaissances du domaine et les connaissances spécifiques au problème, qui sont extraites à partir de solutions idéales entrées par l'enseignant en amont.

Parmi les tuteurs à base de contraintes, nous pouvons citer Kermit [SM02] [SM02]. Ce tuteur porte sur les cours de base de données et permet à l'étudiant de construire des diagrammes ER (Entités-Relation) à l'aide d'une interface WIMP (Windows Icons Menus Pointer). Pour chaque problème, le système enregistre en amont une solution idéale (construite par l'enseignant, ou le programmeur). L'évaluation de la production de l'étudiant se fait en vérifiant les contraintes du domaine non satisfaites dans la solution (la base de connaissance contient 92 contraintes syntaxiques modélisant la connaissance déclarative du domaine). De plus, la solution produite par l'élève est comparée à la solution "idéale" pour vérifier les contraintes sémantiques relatives spécifiquement au problème donné en entrée. C'est cette combinaison entre connaissance du domaine formalisée par les contraintes syntaxiques et connaissance spécifique au problème modélisée par la solution de l'enseignant qui semble bien adaptée à notre problématique de géométrie de construction. Parmi les tuteurs CBT, [Hir+09a] a attiré notre attention du fait qu'il prend en entrée un dessin réalisé par un étudiant sur tablette stylet, pour l'interprétation et la critique de schémas de fournitures. La connaissance du domaine est modélisée par une base de 63 contraintes qui permet d'évaluer la production de l'étudiant. Le système ne dispose néanmoins pas de technique pour la reconnaissance des tracés utilisateur et se contente d'identifier les coordonnées cartésiennes et de générer un modèle 3D. Parmi les CBT, certains s'intéressent à la notion de problème de construction, c'est le cas de Termo-tutor [Mit+11]. Les auteurs proposent une méthodes basée sur les contraintes pour la résolution de problèmes de construction de cycles en thermodynamique. Les auteurs insistent ici sur l'importance de simuler le plus possible l'approche traditionnelle papier-crayon pour garantir la transférabilité de l'apprentissage. Cependant, le dessin des cycles thermodynamiques se fait aussi à partir d'une interface WIMP.

Avec l'évolution des supports numériques et la démocratisation des tablettes orientées stylet, le continuum entre papier-stylo et le numérique, nous semble pouvoir être mis en œuvre de manière encore plus efficace que simplement par l'utilisation d'une interface WIMP. Pour ce faire, il est nécessaire de combiner un moteur tutoriel avec un moteur de reconnaissance de forme pour avoir l'expérience la plus intuitive possible.

Au delà des exemples explicités plus haut, cette méthodologie a été appliqués à plusieurs autres domaines. Plusieurs tuteurs ont par exemple été proposés pour l'apprentissage de la programmation [PR06], [Hir+09b], [HMM09], [GGC09], des bases de données avec SQl-Tutor [Mit12], ou encore de mathématiques discrètes [BR05]. Parmi les tuteurs s'intéressant à la notion de construction, nous pouvons aussi citer [BM06] qui propose un

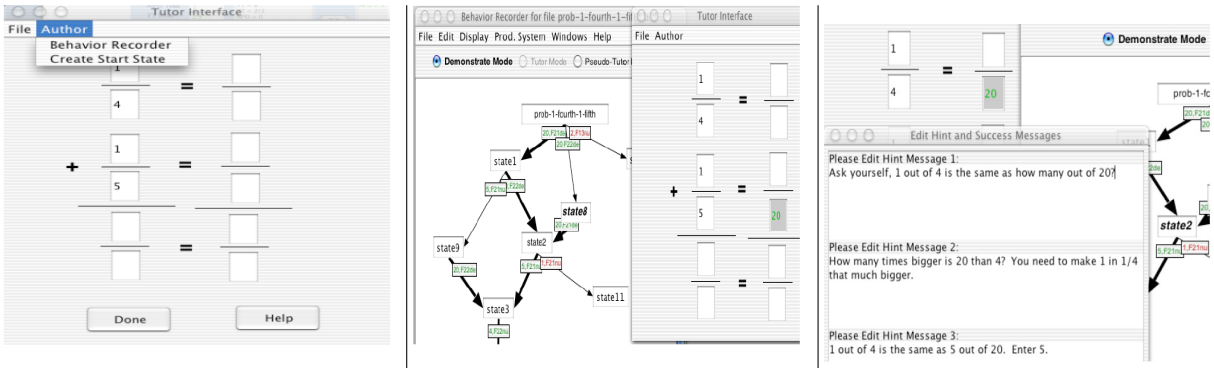
tuteur pour la construction de diagramme UML, avec la spécificité d'intégrer le travail collaboratif entre étudiants.

Une limite de cette famille de tuteurs est l'incapacité à fournir des feedback de guidage. Puisque la connaissance procédurale n'est pas représentée (pas de module expert, pas de règles expertes), un tuteur de ce type n'est pas capable de synthétiser des stratégies de correction à partir de l'état de résolution de l'élève et de les traduire en aides pour l'élève qui serait bloqué à une étape de son raisonnement.

3.2.3 Tuteurs à base de traçage d'exemples

Cette famille de tuteurs est née du constat que le développement de tuteurs est une tâche coûteuse pour les programmeurs. L'idée est de transférer le fardeau de création de contenu pédagogique des programmeurs aux enseignants. Contrairement aux MTT (Model Tracing Tutors) et aux CBT (Constraints Based Tutors), les tuteurs à base de traçage d'exemples ne se basent pas sur une modélisation à priori de la connaissance. Cette famille de tuteurs se contentent d'interpréter les actions des élèves à partir de modèles de connaissances du domaines, générés à partir d'exemples de solution annotés en amont par les enseignants.

Dans [Ale+09] les auteurs explicitent ce paradigme qui se base sur une outil de création de contenu (Authoring Tool), nommé CTAT (Cognitive Tutor Authoring Tool). CTAT permet aux enseignants, ou plus généralement aux non programmeurs, de créer des exercices. Le processus de création de contenu pédagogique dans l'outil CTAT, illustré dans la figure 3.5, réside dans la création par l'enseignant de l'interface graphique du problème (en ajoutant des images, boutons, champs de texte...). Dans la figure 3.5.a, l'enseignant créé une interface de résolution de problème arithmétique, et y ajoute les deux fractions à additionner. C'est cette interface qui va être affichée à l'élève. Ensuite l'enseignant va démontrer sa solution sur l'interface, ce qui a comme conséquence la génération d'un "graphe de comportement" (behavior graph), comme illustré dans la figure 3.5.b. Ce graphe représente les étapes réalisées par l'enseignant pour arriver à une solution, les noeuds étant les états de résolution, et les arcs les actions réalisées. Un chemin dans le graphe représentant une résolution, il est nécessaire que l'enseignant réalise des solutions alternatives pour que le graphe contiennent le plus de chemins de résolutions possibles. Pour les tâches simples, l'enseignant peut ajouter la notion d'ordre partiel entre les étapes, ce qui permet plus de liberté à l'élève. L'enseignant réalise aussi sur l'interface ce qui s'apparente à des erreurs de compréhensions communes (l'équivalent des règles erronées dans les MTT). Enfin, il



a) Création de l'interface du problème

b) Démonstration du problème

c) Ajout des feedback

FIGURE 3.5 – Exemple de création d'exercices avec l'outil CTAT [Koe+04]

peut annoter le graphe par des feedback de corrections qui seront affichés à l'élève (*cf.* figure 3.5.c).

Cette approche à base de traçage d'exemple a été adaptée à plusieurs domaines, tels que les mathématiques ([Ear14], [AMS09], [McL+15]), ou la programmation [Jin+12] [Jin+14].

Dans la même veine que CTAT, Assistements [Hef+06] est un système tutoriel intelligent générique (*i.e.* indépendant du domaine) qui permet de créer des contenus pédagogiques par traçage d'exemples, et qui a été adapté à plusieurs domaines tels que la physique, les sciences de la vie et de la terre, ou les statistiques [Gob+10] [HH14].

Cette notion de système tutoriel intelligent générique a fait l'objet de plusieurs travaux, nous pouvons citer dans ce cadre le système SEPIA [Gin+14]. L'intérêt de ce système est qu'il peut être intégré en tant que "plug-in" dans des applications existantes qui ne contiennent pas d'assistance pédagogique. C'est au concepteur ou à l'enseignant de spécifier les scénarios d'interventions du tuteur, par un ensemble de règles qui modélisent d'une part une actions possible de l'utilisateur, et d'autre part le feedback à déclencher une fois cette action effectuée. La différence de SEPIA par rapport aux tuteurs à base de traçage d'exemples est qu'il n'y a pas de modélisation de la connaissance du domaine à partir de la spécification de l'enseignant, ce qui limite les possibilités en termes d'analyse fine et d'évaluation poussée des actions de l'élève.

Une limite évidente du paradigme de tuteurs à bases d'exemples si on voudrait l'adapter à notre problématique d'apprentissage de la géométrie réside dans le fait qu'il est compliqué pour l'enseignant de représenter toutes les erreurs de construction que pour-

rait commettre l'élève. Cette limite est similaire à celle que nous avions dégagée pour les tuteurs à base de règles. Ajoutons qu'il est tout aussi compliqué pour l'enseignant de réaliser **toutes les solutions** d'un problème de géométrie, surtout s'il est complexe.

Toutefois, la notion de mode auteur dans les CTAT, et plus généralement dans les systèmes tutoriels génériques, qui consiste à générer des exercices à partir de réalisations produites par les enseignants nous semble pertinente, et nous nous en inspirerons pour la conception d'un mode auteur dans notre système. En effet, la création de problèmes à partir d'**un exemple unique de solution** dessiné par l'enseignant directement sur l'interface nous semble être un élément facilitateur de l'usage du système.

3.2.4 Approches basées sur les données

On a vu dans les sections précédentes que la modélisation du domaine expert dans la littérature s'est grandement basée sur une ingénierie des connaissances, par règles de production ou par contraintes, ce qui entraîne un coût de développement élevé. L'approche basée sur les exemples est une première piste pour résoudre ce problème. Une approche plus récente tient compte de la large quantité de données de résolutions de problèmes des élèves, notamment dans les MOOC (*Massive Open Online Courses*). Dans un tel contexte (de disponibilité de données), l'objectif est de réussir à générer des feedback adaptatifs en comparant la solution d'un élève à l'historique des solutions des autres apprenants en essayant de trouver des similarités. Dans la littérature, bien que cette approche soit largement suivie pour l'apprentissage de la programmation, elle l'est peu pour les autres domaines STEM (Science, Technology, Engineering, and Mathematics).

Dans [BS08], les auteurs présentent une technique nommée "Hint Factory", qui consiste à générer des feedback automatiquement à partir de données antérieures d'exercices d'étudiants en logique inductive. Chaque réalisation de preuve logique est représentée par un graphe solution, où les noeuds représentent les états de résolution, et les arcs les actions réalisés par les élèves. Tous les graphes solutions sont fusionnés en un seul graphe, qui représente donc toutes les stratégies que les élèves ont suivi pour élaborer une preuve. Ce graphe est ensuite transformé en processus de décision markovien (MDP *Markovian Decision Proocess*). Un MDP est défini par un état S, un ensemble d'action A, des probabilités de transition P, et une fonction de récompense R. Un apprentissage par renforcement (en paramétrant la valeur de récompense de l'état final (état solution) à une valeur maximale, et une fonction de coût uniforme pour chaque action) est effectué, pour trouver la solution optimale du problème dans le MDP. Cela permet aussi d'assigner des valeurs

de récompenses pour tous les états du graphe. Quand un étudiant demande de l'aide, le tuteur met en correspondance son état de résolution avec un états du MDP, et le feedback retourné consiste à communiquer la meilleure transition sortante de cet état. Cette approche a été adaptée à d'autres domaines, tels que l'apprentissage des listes chaînées [Fos+09], le tutorat SQL [LMZ16], ou les environnements d'apprentissage basique de la programmation [IHB14]. Toutefois, une limite de cette approche est la complexité. En effet, pour des espaces de solutions larges, il est difficile de mettre en correspondance toutes les données des élèves et il se peut qu'il n'y ait pas d'états correspondant à celui de l'élève qui demande de l'aide. Par exemple, dans [BS08], pour le domaine assez restreint de logique inductive, le tuteur a réussi à générer un feedback dans seulement 81% des cas de demande d'aide d'étudiants.

D'autres approches plus récentes se basent sur l'apprentissage profond pour générer des feedback et se concentrent sur la correction de codes erronés dans le contexte de l'apprentissage de la programmation. Dans [BS16], les auteurs utilisent une large base de soumissions correctes d'étudiants pour résoudre un problème de programmation donné pour entraîner un réseau de neurones récurent [RM87]. L'objectif est d'apprendre un modèle qui permet de corriger des erreurs de syntaxe en programmation. Toutefois, le modèle n'est capable de corriger les erreurs que dans 31,9% des cas.

Dans la même veine, les auteurs de [Gup+17] proposent DeepFix, qui se base sur un réseau de neurones multi-couches avec attention [BCB14], pour localiser des erreurs dans des programmes en C et de proposer des corrections. Sur un ensemble de 6971 programmes erronés réalisés par des étudiants pour un total de 93 exercices, le système n'a pu corriger que 27% des programmes.

Il nous semble donc que ces approches, bien qu'intéressantes, ne semblent pas encore assez mures pour concurrencer la performance des méthodes traditionnelles, telles que les CBT ou les tuteurs à base de règles qui ont fait leurs preuves.

3.3 Systèmes tutoriels dédiés à l'apprentissage de la géométrie

Nous nous intéressons maintenant aux systèmes éducatifs dédiés à la géométrie. D'après notre étude de la littérature, peu de travaux portent sur la géométrie de construction. La plupart des approches concernent plutôt la démonstration de preuves en géométrie comme Mentoniezh [Py01], un tuteur pour la démonstration de preuves en géométrie. Dans ce

système, les figures ne sont pas manipulées. La représentation de la connaissance se base sur de la logique de premier ordre, avec le langage HDL, pour Hypothesis Description Langage, qui permet de modéliser les théorèmes de géométrie de niveau collège sous forme de règles. Ce système est divisé en deux composants majeurs : le module expert qui est une fusion du module du domaine et du module de tutorat, et le module de l'apprenant. Le *module expert* est donc responsable de la synthèse de strategie tandis que le *module de l'apprenant* est responsable de l'interprétation et de la supervision en temps-réel des actions de résolution de l'élève. Les auteurs considèrent le problème de synthèse de stratégies en tant que *problème de planification* [MH69], tel que le but est d'appliquer une séquence d'actions du domaine (*théorèmes, faits..*). Dans ce système, la base de connaissance est un ensemble de règles de type si-alors, qui représentent les axiomes géométriques et les théorèmes correspondant au niveau collège. Le module expert applique cette connaissance sur les variables d'un problème donné. Le module de l'apprenant interprète les actions de l'élève (*i.e.* les étapes de la démonstration) en mettant en correspondance le plan que l'élève a choisi, en temps réel, avec les solutions générées par le module expert. La figure 3.6 illustre la représentation de la connaissance pour un problème de démonstration avec le tuteur Mentoniezh. L'un des points forts de ce système réside dans sa capacité à reconnaître le plan, ce qui permet la génération de feedback et le guidage de l'élève. Même si notre domaine d'application, la construction de figures géométriques, est différent de la démonstration de preuves, **L'architecture de notre moteur de supervision s'inspire de ce travail**.

Hypothèses ABIJ parallélogramme **et** (AI) $\perp$ (BJ)
Théorème Si un parallélogramme a deux diagonales perpendiculaires, alors c'est un losange
Conclusion ABIJ est un losange

FIGURE 3.6 – Exemple de pas de preuve dans Mentoniezh[Py, 2001]

Un autre système tutoriel intéressant pour la démonstration de preuves est QED-Tutrix [FRG18], dont l'interface est illustrée dans la figure3.7. Ce tuteur permet à l'élève de manipuler une figure représentant le problème, en utilisant un plugin Geogebra, pour explorer ses propriétés et ensuite établir ses pas de preuves pour résoudre le problème. La manipulation de la figure est mise en oeuvre dans un but purement exploratoire et n'entre ni dans le processus d'établissement de preuves, ni dans l'évaluation du tuteur. Comme illustré dans la figure 3.7, le processus de création d'exercices y est fastidieux,

car le concepteur doit élaborer la figure, traduire les hypothèses et conclusions en langage Prolog, et construire à la main le graphe solution, comportant tous les pas possibles à réaliser pour arriver à une démonstration correcte. Notons que dans des travaux récents, les auteurs ont proposé l'automatisation de ce processus de construction de preuves [Fon+20].

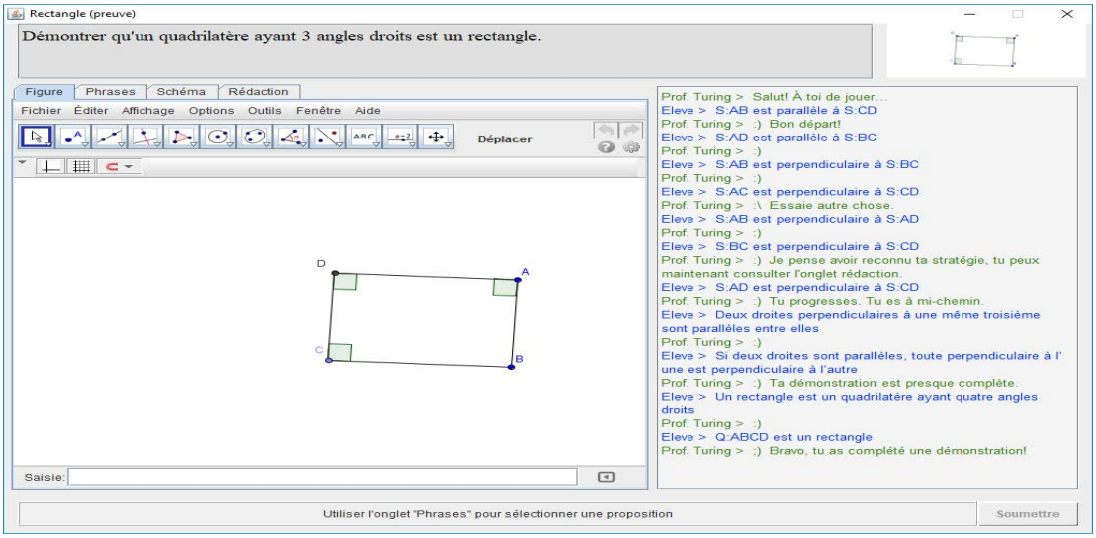


FIGURE 3.7 – Interface du tuteur QED-Tutrix [FRG18]

Plusieurs autres systèmes tutoriels ont été proposés pour l'aide à la démonstration de preuves, en se basant sur des principes similaires aux deux approches précédentes [Bal+03] [MV05]. Certains proposent des méthodes de génération automatique de problèmes de démonstrations à partir de modes auteurs plus ou moins intuitifs. Dans [Alv+14], les auteurs présentent une méthode pour la génération automatique de problèmes de preuves en géométrie à partir de l'analyse d'une figure donnée en entrée par l'utilisateur. Par exemple, dans [SHM14], les auteurs proposent un système pour la génération d'exercices de géométrie de niveau lycée. L'utilisateur peut y sélectionner un ensemble de concepts dans l'interface (*e.g.* triangle + perpendicularité), puis l'algorithme génère une figure relative à ces concepts. À partir de cette figure, un problème de démonstration de preuves est généré. **Notre approche se veut plus directe, plus accessible : nous voulons que l'enseignant puisse générer un exercice de construction en dessinant uniquement un exemple de solution directement sur IntuiGeo**.

Parmi les quelques travaux s'intéressant à la géométrie de construction, nous pouvons mentionner l'article de *Gulwani et al.* [GKT11], dans lequel les auteurs proposent une méthode pour la synthèse de constructions géométriques pour résoudre des exercices ty-

piques, tels que "trace un triangle étant donnée la longueur de ses trois cotés". Les auteurs abordent la résolution de ces exercices en tant que problème de synthèse de programmes [Gul10], qui peut être reformulé en un problème de planification. Le solveur doit être capable de résoudre automatiquement le problème donnée par la synthèse de plans de construction, dont les étapes sont des actions de dessin de type règle-compas, qui mènent à la construction de la figure désirée. Cependant, cette approche se limite à la résolution de problèmes de construction, les auteurs ne proposent pas de tuteur basé sur leur solveur, ni de méthodes de reconnaissance ou d'évaluation d'actions d'un utilisateur/élève. Cette absence de tuteurs dédiés à la résolution d'exercices de construction peut s'expliquer par la complexité du domaine d'application. Alors que pour la démonstration de preuves, il est plutôt facile de définir les contours de la connaissance du domaine (les théorèmes, les hypothèses, et les pas de preuves possibles), pour la construction, un large éventail d'actions de dessins est possible, avec un espace de solutions très large en tenant compte de l'ordre des tracés, et des stratégies de construction variées. Une autre explication pourrait être le fait que la géométrie de construction requiert la reconnaissance des tracés de l'élève. Pour ce faire, un tuteur dédié à ce domaine devra se baser sur une jonction de techniques de reconnaissance de formes avec celles des STI, ce qui n'est pas évident. C'est cette approche de couplage entre ces techniques que nous adopterons pour notre système tutoriel tant elle semble adaptée à notre problématique. Dans la prochaine section, nous nous intéresserons aux STI basés sur l'interprétation des tracés de l'utilisateur.

3.4 Systèmes tutoriels basés sur du dessin

La plupart des approches vues jusqu'à présent se basent sur l'approche WIMP. Nous nous intéressons ici aux travaux qui se rapprochent des nôtres, par les aspects simulation de l'approche papier-crayon et celui du dessin "libre" pour l'apprenant.

Comme nous l'avons vu dans ce chapitre et le précédent, il existe beaucoup d'approches dans la littérature qui abordent l'interprétation de documents structurés en ligne, de même pour les systèmes tutoriels, mais à notre connaissance, peu de travaux considèrent ces deux aspects pour un même système. Vu la complexité de la tâche de reconnaissance, certaines approches combinent les actions de dessin libre et les actions d'annotation. C'est à dire que l'utilisateur dessine ses tracés, qui seront segmentés par le système, mais aussi étiquette les objets segmentés pour que le système puisse les analyser. Ceci leur permet d'être domaine-indépendants : pas besoin d'avoir de connaissance spécifique au domaine puisque

c'est l'utilisateur qui indique quelle entité vient d'être tracée [BJ13] [For10]. Cela permet certes d'éviter d'éventuelles erreurs de reconnaissance, mais en contrepartie entrave la fluidité du processus de dessin de l'utilisateur.

Pour la géométrie de construction, comme nous l'avons mentionné dans la section précédente, il n y a pas à notre connaissance de tuteurs dédiés à ce domaine. La plupart des approches se contentent de proposer une méthode de reconnaissance de tracés en ligne dénuée de technique d'analyse sémantique et de génération de feedback [Cos+11] [CGJ12]. Nous allons voir les approches existantes dans les domaines du STEM.

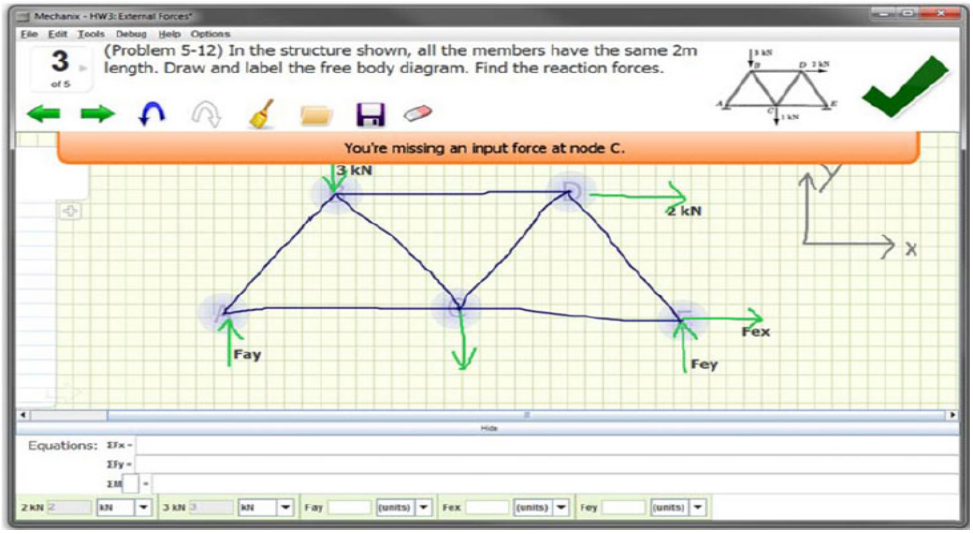


FIGURE 3.8 – Interface du tuteur Mechanix [Ati+14]

Par exemple, le tuteur Mechanix [Ati+14], a été développé comme support pour l'apprentissage interactif de cours de mécanique et il permet en particulier l'interprétation de croquis. Pour la partie reconnaissance en ligne de schémas, Mechanix se base sur le framework d'analyse de tracés Paleosketch [PH08], présenté dans la section 2.3.3. Comme illustré dans la figure 3.8, l'étudiant dessine librement des treillis (segments en bleu) et les complète par des vecteurs de force (flèches vertes). Au delà de la reconnaissance des treillis, le système compare le dessin de l'étudiant à un dessin de référence fourni par l'enseignant (les deux treillis sont représentés par des graphes). Le système dispose d'un solveur qui est capable de synthétiser des équations à partir du dessin, pour ensuite vérifier si elles satisfont les contraintes de l'exercice spécifiées en amont par l'enseignant. Le feedback est généré une fois que l'élève soumet sa réalisation, il est donc dit *a posteriori*. L'analyse du déploiement de cette solution en classe a permis de démontrer l'impact positif du tuteur sur les étudiants l'ayant utilisé par comparaison à ceux qui ont travaillé

avec la méthode traditionnelle papier et crayon. L'approche est donc intéressante, cependant la génération des feedback a posteriori (après que l'élève ait validé son schéma) nous semble peu adaptée puisque l'on veut pouvoir guider l'élève dans son tracé, notamment si il est bloqué à une étape de sa réalisation. Pour ce faire, il faudrait que le feedback soit produit à la volée et en temps-réel. De plus, l'aspect tutoriel consistant uniquement à extraire un ensemble d'équation du schéma dessiné par l'élève limite un peu les capacités tutorielles de Mechanix (le système n'est pas capable de donner des pistes de guidages sur les prochaines étapes).

Parmi les systèmes basés sur l'analyse des tracés, citons ceux dédiés à l'apprentissage de l'écriture [Sim+18], ou de la musique [TBH15]. Ces systèmes ont comme point commun l'absence de module expert. L'analyse des réponses des élèves se base sur la reconnaissance des tracés couplée à une comparaison de la production à un schéma de référence de l'enseignant. Cela s'explique surtout par le fait que dans ces domaines, il n'y a pas réellement de protocole/ d'étapes de résolution comme il en existe pour construire une figure ou pour démontrer une preuve. Bien que les systèmes cités ont pu démontrer un impact positif sur la performance des élèves [Bon+20], nous voulons que notre tuteur en géométrie soit capable, en plus de fournir un feedback correctif à la manière des systèmes cités plus haut, de pouvoir donner un feedback de guidage sur les prochaines étapes à réaliser quand l'élève est bloqué, à la manière d'un système tutoriel intelligent classique.

Dans cette idée, P. A. Jacques *et al.* [Jaq+13] ont intégré une brique de reconnaissance de tracés à un tuteur classique WIMP appelé Pat2Math [Mor+17]. Pat2Math est un système d'apprentissage de l'algèbre basé sur la technique de traçage de modèle (règles expertes et règles erronées). La brique de reconnaissance est basée sur un SDK développé par l'entreprise myScript [1]. Une application web couplant ces deux techniques a été développée. L'idée de cette intégration est de voir l'impact qu'aurait le dessin libre sur la performance de l'élève, l'hypothèse étant que la réduction de la charge cognitive induite par le dessin à main levée induirait de meilleurs résultats. Toutefois, dans leur étude, les auteurs rapportent qu'il n'y a pas de différence significative entre deux groupes d'étudiants, le premier utilisant le tuteur classique, et le second le tuteur enrichi par la reconnaissance des tracés. Les auteurs expliquent ce résultat contre-intuitif par le fait que la condition expérimentale, spécifiquement la connexion wifi trop lente pour les tablettes, rendait les processus de reconnaissance et de génération de feedback trop lents, ce qui était contre-productif pour les élèves. Nous pouvons en conclure qu'**il ne suffira**

1. `http://myscript.com/technology/#math`

pas que notre tuteur soit capable d'analyser les tracés à la volée en plus de générer des feedback, il faudra qu'il le fasse rapidement afin de rendre ces processus transparents à l'apprenant (moins d'une seconde).

3.5 Discussion et bilan

Dans ce chapitre, nous nous sommes intéressés à la littérature des STI en étudiant les grandes familles d'approches. En ce qui concerne les tuteurs à base de règles, nous avons vu les avantages offerts par la présence d'un module expert, en particulier la capacité de modéliser la connaissance procédurale et de guider l'élève dans sa réalisation. Cependant, cela implique la définition de règles erronées modélisant toutes les erreurs possibles, en plus de la génération de tous les plans solutions valides. Ce processus s'avère fastidieux pour les domaines d'application complexe tel que la géométrie de construction qui se caractérise par une grande liberté d'action de l'élève. Aussi, appliquer cette approche pour évaluer le processus de réalisation s'avère compliqué, surtout en tenant compte de la contrainte d'interaction et de génération de feedback en temps-réel. Les tuteurs à base de contraintes (CBM) évitent ce problème en proposant une méthode d'évaluation basée sur l'état de résolution, et non pas sur le processus, en modélisant la connaissance par les contraintes. La génération du feedback consiste alors à examiner la validité de l'action de l'élève en vérifiant que sa production ne viole pas les contraintes du domaine. Cependant, l'absence de module expert dans ce type de tuteur implique l'incapacité de synthétiser des stratégies de résolution expertes permettant de guider l'élève et de générer des conseils sur les prochaines étapes à réaliser, ou sur les actions rétro-actives à effectuer par l'élève pour corriger ses erreurs.

Pour profiter des avantages qu'offrent ces deux familles de tuteurs tout en essayant d'éviter leur limites, nous proposons une approche originale que l'on pourrait qualifier d'hybride, au sens où nous combinons certaines caractéristiques des deux familles pour pouvoir évaluer les productions en temps-réel tout en étant capable de synthétiser des stratégies de résolution. Ce système original conçu et développé pendant cette thèse, dénommé IntuiGeo, pourra s'apparenter à un tuteur à base de contraintes au sens où la connaissance spécifique à chaque problème sera représentée par un graphe de connaissance. Ce graphe contiendra les objets géométriques et les contraintes correspondantes à ces objets. Ces informations seront extraites d'un exemple de solution dessiné par l'enseignant. Notre module de l'apprenant interprétera sémantiquement les éléments géométriques reconnus

à partir des tracés manuscrits de l'élève et les mettra en correspondance avec un des éléments du graphes pour évaluer les contraintes satisfaites et violées. Cette interprétation permet de générer du feedback correctif à la manière d'un tuteur CBM. D'autre part, IntuiGeo pourra s'apparenter à un tuteur à base de règles au sens où l'on y inclut un module expert. La connaissance procédurale est définie dans ce module par un environnement de planification dynamique, avec des actions de dessins de type règle-compas qui simuleront le comportement de l'expert lors de la résolution d'un exercice de construction. À la demande de l'élève, le module expert sera capable de générer des stratégies de résolution à partir de l'état de sa figure et de transformer le plan trouvé en feedback de guidage. De plus, nous proposons un mode auteur intuitif, en nous inspirant des tuteurs à base d'exemples, qui permet à l'enseignant de créer un nouveau problème simplement en dessinant un exemple solution sur l'interface. La différence ici est que l'enseignant ne sera pas obligé d'annoter sa solution ou de fournir tous les plans solutions possibles, cela se fera automatiquement. Enfin, une autre originalité de notre approche vient du fait qu'elle combine pour la première fois des techniques de reconnaissance de formes et des techniques venant des STI pour concevoir un système tutoriel intelligent, orienté stylet, et dédié à la géométrie de construction.

3.6 Préambule de la suite du manuscrit

A la fin de cette partie état de l'art, nous pouvons donc présenter l'architecture de notre système tutoriel orienté stylet. Celle-ci est illustrée dans la figure 3.9. La combinaison

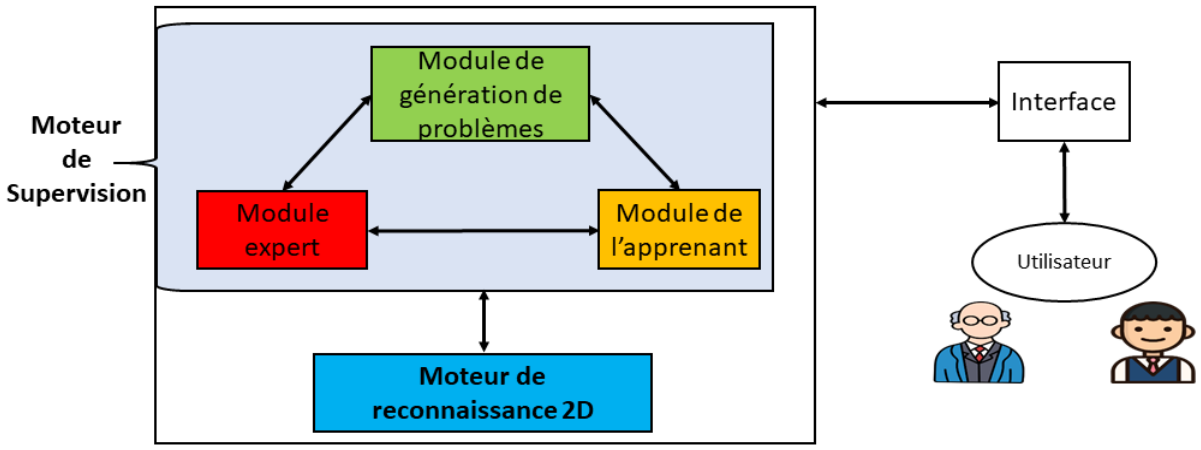


FIGURE 3.9 – Architecture d'IntuiGeo

entre reconnaissance de formes et tutorat y est clairement transparente. En effet, notre système se base sur deux moteurs :

- **Un moteur de reconnaissance 2D** : encapsule la connaissance relative à l'interprétation de formes manuscrites à la volée ;
- **Un moteur de supervision** : représentant l'aspect tutoriel du système, interagissant avec le moteur de reconnaissance 2D, et défini par trois modules :
  - *Un module de génération de problèmes* qui permet à l'enseignant de créer des exercices à partir d'un exemple de solution dessinée sur l'interface et interprétée par le moteur de reconnaissance 2D ;
  - *Un module de l'apprenant* qui est responsable de la représentation de l'état de résolution de l'élève, et donc de l'évaluation de ses actions. Chaque tracé, interprété par le moteur de reconnaissance 2D, est ensuite analysé et évalué relativement aux contraintes de l'exercice à résoudre ;
  - *Un module expert* capable de synthétiser des stratégies de guidage/correction à partir de l'état de résolution de l'élève, s'il est bloqué dans sa résolution.

Dans la prochaine partie, nous présentons le premier axe de nos contributions, qui consiste en l'extension du formalisme grammatical GMC-PC pour l'interprétation à la volée et *en temps-réel* de tracés manuscrits. Ce premier axe de nos contributions vise à permettre à notre *moteur de reconnaissance 2D* de respecter les contraintes d'interaction en temps-réel avec l'utilisateur, en vue de générer des feedback immédiats.

Dans la troisième partie, nous présentons le deuxième axe de nos contributions, relatif à l'aspect tutoriel de notre système, c'est à dire le moteur de supervision.

Deuxième partie

Interprétation à la volée de schémas géométriques

Chapitre 4

GMC-PC : Grammaire Multi-ensembles à Contraintes Pilotée par le Contexte

Dans notre étude bibliographique, nous nous sommes intéressés aux différentes approches pour l'interprétation à la volée de schémas manuscrits structurés. Nous avons vu que les approches structurelles permettaient de modéliser les dépendances spatiales et les contraintes spatiales entre différentes formes qui composent le document. Par contraste, les approches statistiques sont plutôt adaptées à la reconnaissance de symboles isolés mais ne permettent pas de modéliser explicitement la structure du document. Les approches hybrides, qui combinent interprétation globale structurelle et reconnaissance statistiques, apparaissent donc particulièrement adaptées à la reconnaissance de figures géométriques.

L'objectif de ce premier axe de thèse est d'élaborer une méthode d'interprétation à la volée de schémas manuscrits structurés qui puisse offrir une composition interactive en temps-réel avec l'utilisateur. Il est donc important de contrôler la complexité du processus d'analyse au fur et à mesure que le document se complexifie.

Nous avons choisi de nous baser sur le formalisme GMC-PC (Grammaire Multi-ensembles Pilotées par le Contexte) et l'analyseur associé pour la modélisation de la connaissance a priori du domaine et l'interprétation incrémentale des tracés de l'utilisateur.

Cette grammaire est générique et a été adaptée avec succès auparavant à plusieurs domaines d'applications [Gho12] [PMA10] [MA09]. Elle est donc assez expressive pour être adaptée au domaine de la géométrie. Comme nous le verrons par la suite, en l'état, l'interprétation de figure complexe par cette approche peut conduire à un problème d'explosion combinatoire (dessin de polygones complexes, dessin de sous-figures à partir de figures existantes).

Il a donc fallu enrichir et étendre ce formalisme pour contenir la combinatoire et

respecter la contrainte d'interaction temps-réel avec l'utilisateur. Cette contribution porte sur l'extension du formalisme en garantissant sa généricité et on ouvrant ainsi son potentiel à la composition de schémas manuscrits structurés complexes, comme peuvent l'être les figures géométriques, mais aussi d'autres domaines, comme les plans d'architecture par exemple.

Avant de présenter cette contribution, nous nous intéressons dans ce chapitre au formalisme de base, proposé par [MA09]. Nous définissons formellement la grammaire, le processus d'analyse associé, et les limites du formalisme par rapport à notre domaine d'application et à nos besoins. Nous présenterons à la fin de ce chapitre les pistes qui vont ensuite permettre l'extension de la grammaire et l'amélioration de la performance en terme de complexité d'analyse.

4.1 Définitions et principes généraux

Definition 4.1.1. Une Grammaire Multi-ensembles Pilotées par le Contexte (GMC-PC) [MA09] est définie formellement par un tuple G=(V_N, V_T, S, P) avec :

- V_N : l'ensemble des symboles non terminaux = les classes d'éléments (segment, cercle, triangle, etc) ;
- V_T : l'alphabet, ici $V_T = \{tracé\}$;
- S : l'axiome de départ ;
- P : l'ensemble des règles de production.

Une règle de production $p \in P$ est définie par :

$$\alpha \rightarrow \beta \left\{ \begin{array}{l} \text{Préconditions} \\ \text{Contraintes} \\ \text{Postconditions} \end{array} \right\} \mid \alpha \in {V_N}^+,\ \beta \in (V_T \cup V_N)^+$$

Une règle de production dans le formalisme GMC-PC consiste à réduire un ensemble d'éléments β en un ensemble d'éléments α si un ensemble de conditions est satisfait. La modélisation sous formes de blocs (Préconditions, Contraintes, Postconditions) permet de formaliser deux visions du documents : une vision globale sur le contexte et la structure par les blocs de préconditions et postconditions tandis que le bloc de contraintes modélise une vision locale du tracé analysé.

Les préconditions et les postconditions sont basées sur le concept de Contexte Structu-

rel du Document (CSD) qui modélise une zone dans le document et les éléments attendus dans cette zone, défini par :

Definition 4.1.2. Un CSD s'écrit sous la forme $(\lambda)[position](\gamma)[partie]$ avec :
- λ : l'élément de référence ;
- *position* : une zone (*i.e* une position) relative à λ ;
- γ : l'ensemble des éléments attendus dans cette zone ;
- *partie* : une partie des éléments attendus (premier point par exemple).

Les préconditions sont un ensemble de CSD qui doivent être satisfaits pour que la production soit applicable. Les postconditions sont des règles spéciales dont la prémisse est un ensemble de CSD et la conclusion un type de production à déclencher si les CSD en prémisse sont satisfaits. Cela permet de prédire les objets qui pourraient être créés à partir des nouveaux éléments α. Cette formalisation permet de piloter le processus d'analyse par le contexte. En effet, les préconditions représentent l'étape de vérification tandis que les postconditions représentent l'étape de prédiction.

4.2 Modélisation de la connaissance par les règles de productions

Pour préciser le rôle et le fonctionnement de chaque bloc, nous nous appuyons sur l'exemple de création d'un angle dont la règle de production est présentée sur la figure 4.1, et les étapes de dessin sont présentées dans la figure 4.2. Le tracé t (en rouge dans 4.2) va être réduit en angle *res* si un ensemble de conditions sont satisfaites. Les trois blocs représentent les trois étapes mises en oeuvre pour la réduction d'une règle de production que nous présenterons dans les sous-sections suivantes : l'étape de *vérification*, l'étape de *reconnaissance*, et l'étape de *prédiction*.

4.2.1 Étape de vérification : le bloc de préconditions

Premièrement, l'analyseur vérifie la validité du contexte structurel du tracé par le biais du bloc de préconditions. Ce bloc modélise une vision globale du tracé sans pour autant considérer sa forme. Les préconditions sont un ensemble de CSD qui doivent être satisfaits et représentent le contexte cohérent qui permet de remplacer β par α. La fonction de ces CSD est de vérifier que l'élément de la partie droite de la règle (ici le tracé t) est

Angle : res → tracé : t avec :

Préconditions :
(Segment :s1)[LongueurSegment] (t)[Premier] &
(Segment :s2)[LongueurSegment] (t)[Dernier]

⟶ (a) Le bloc de préconditions modélise **la verification** d'un contexte cohérent

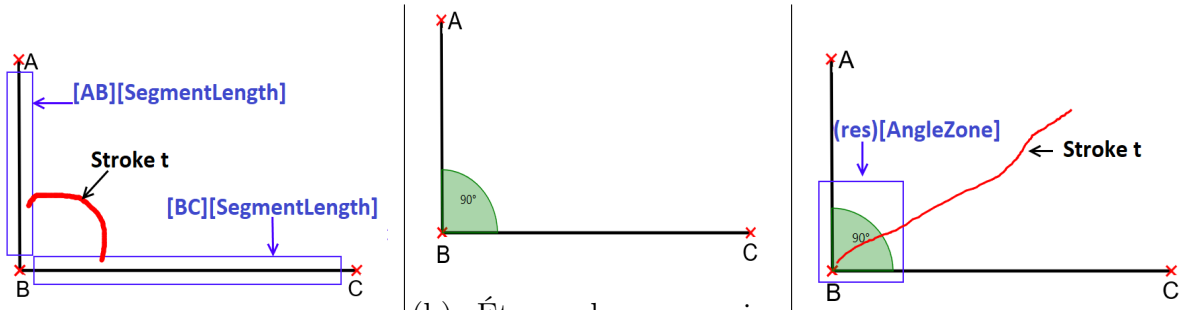
Contraintes :
Reconnaisseur(t, Arc)
Adjacents(s1, s2)

⟶ (b) Le bloc de contrainte **reconnaît** la forme de la trace et vérifie des contraintes structurelles

Postconditions
(res)[AngleZone] (tracé : t') [Premier] ⟹ [Bissectrice → t']

⟶ (c) Les postconditions sont de nouvelles zones permettant **la prédiction** de nouveaux objets et la mise à jour du document

FIGURE 4.1 – Règle de création d'un angle (res) en GMC-PC à partir d'un tracé manuscrit t intersectant deux segments existants (s1 et s2)



(a) Étape de vérification

(b) Étape de reconnaissance

(c) Étape de prédiction

FIGURE 4.2 – Rôles des blocs de la production correspondant aux trois étapes de la réduction d'une production : a) étape de vérification d'un contexte cohérent, b) étape de reconnaissance de la forme du tracé, c) étape de prédiction et de mise à jour de la structure du document

structurellement cohérent avec la production. C'est cela qui rend le formalisme sensible au contexte global du document.

Dans le bloc de préconditions de la figure 4.1, le CSD :

(Segment :s1)[LongueurSegment] (t)[Premier]

signifie que le premier point du tracé t doit appartenir à une zone recouvrant la longueur d'un segment quelconque s1 (*LongueurSegment*, *cf.* figure 4.2.a). La même contrainte est spécifiée dans ce bloc en mettant en jeu le dernier point du tracé et un autre segment quelconque s2. Notons ici que la référence à des éléments déjà contenus dans le document (*i.e.* s1 et s2) dans les CSD permet de les utiliser dans la règle sans pour autant les transformer en d'autres éléments. Si le contexte structurel est satisfait, l'analyseur passe au bloc de contraintes.

4.2.2 Étape de reconnaissance : le bloc de contraintes

Les contraintes modélisent une vision locale sur les éléments analysés β. Nous distinguons deux types de contraintes : les *contraintes statistiques* qui permettent de reconnaître la forme du tracé, et les *contraintes structurelles* qui permettent de vérifier l'agencement spatial des éléments entrants en jeu dans la production.

Dans l'exemple de la figure 4.1, on remarque la présence de ces deux types de contraintes :

— une contrainte statistique, la contrainte *Reconnaisseur(t, Arc)*, qui stipule que le tracé t doit être assimilé à un arc ;

— une contrainte structurelle, la contrainte *Adjacents(s1, s2)*, qui stipule que les deux segments s1 et s2 doivent être adjacents.

Si le bloc de postconditions et le bloc de contraintes sont satisfaits, la production est réduite (*i.e.* le nouvel élément angle est créé), et des nouveaux CSD sont créés *via* le bloc de postconditions.

4.2.3 Étape de prédiction : les postconditions

Le rôle de ce bloc est de mettre à jour la structure du document, par la création de nouvelles zones de contact par le biais des CSD. Une postcondition est de la forme :

$$(\lambda)[position](\gamma)[partie] \implies [\alpha \leftarrow \beta].$$

La partie gauche de cette formalisation reprend la forme des CSD telle que définie auparavant et modélise la création d'une nouvelle zone dans le document, relative au nouvel élément créé. La partie droite [$\alpha \leftarrow \beta$] modélise le format de production à déclencher par l'analyseur si le nouveau CSD créé est satisfait. Dans l'exemple de la figure 4.1, le CSD

$$\text{(res)[AngleZone] (tracé : t') [Premier]} \implies \text{[Bissectrice} \rightarrow \text{t']}$$

modélise la création de la nouvelle zone AngleRes recouvrant l'angle *res* créé par la production (*cf.* figure 4.2.c). Le bloc de postconditions modélise donc l'étape de **prédiction** de la création de nouveaux éléments et la mise à jour de la structure du document. C'est cette formalisation qui permet de **guider le processus de l'analyse par le contexte**.

4.3 Processus d'analyse

Nous nous intéressons dans cette section au processus d'analyse associé au formalisme GMC-PC. Dans [MA09], Macé présente en détail les principes de l'analyseur associé au formalisme GMC-PC. La figure 4.3 illustre ce processus de façon globale. L'analyseur du formalisme GMC-PC s'inspire, logiquement, de l'analyseur de la grammaire GMC [Mar94]. Le processus d'analyse des GMC consiste à appliquer systématiquement les règles de productions dont les contraintes sont satisfaites pour remplacer des multi-ensembles d'éléments β par des multi-ensembles d'éléments α, jusqu'à arriver à stabilité, c'est à dire qu'il n'y a plus de productions à réduire.

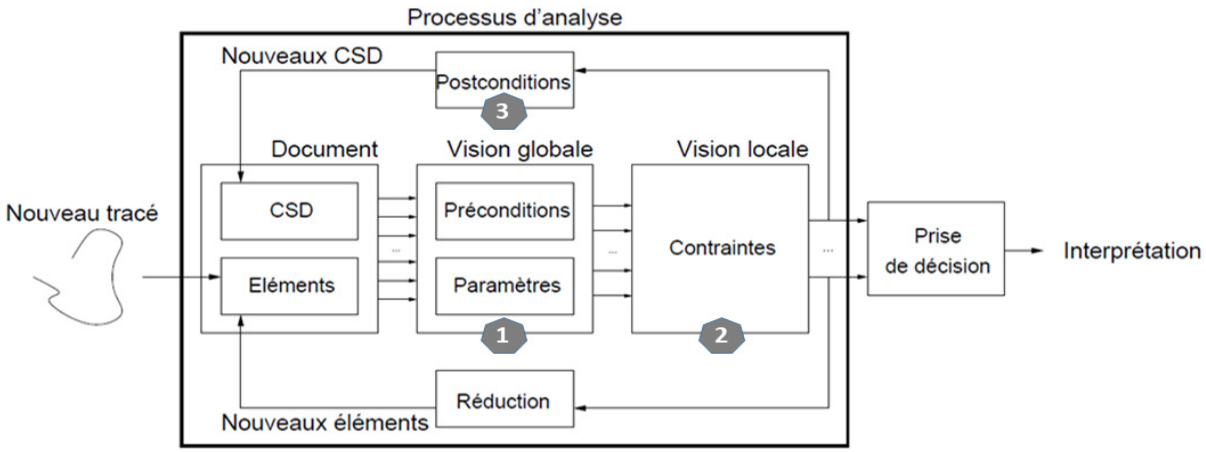


FIGURE 4.3 – Schéma de principe du processus d'analyse [MA09]

4.3.1 Analyse guidée par le contexte

La définition des blocs de préconditions et de postconditions dans le formalisme GMC-PC a pour conséquence que seules les productions structurellement cohérentes sont testées par l'analyseur, ce qui réduit l'espace de recherche de façon significative. Le processus est un couplage entre une stratégie d'analyse ascendante (guidée par les nouveaux éléments) et descendante (guidée par les CSD créés en postcondition). Pour chaque nouvel élément (tracé, segment, angle...), l'analyseur recherche les CSD qu'il satisfait (*i.e.* les zones de contact auxquelles il appartient). De même, pour chaque nouveau CSD créé en postcondition, l'analyseur recherche les éléments qui le valident, parmi tous les éléments du document. Par conséquent, on aura des couples (élément, CSD) qui modélisent des relations structurellement cohérentes. Ces couples sont le point de départ du déclenchement des règles de productions, et seules les règles *contextuellement cohérentes* seront déclenchées. Une production est contextuellement cohérente pour un couple (élement, CSD) si elle contient l'élément en question dans sa partie droite (*i.e.* en paramètre), et le CSD correspondant dans son bloc de préconditions. Considérons l'exemple illustré dans la figure 4.2.c, avec le couple (élément, CSD) correspondant à

(**CSD** : (res)[AngleZone] (tracé : t') [Premier] $\Longrightarrow$ [Bissectrice $\rightarrow$ t'], **élément** : t').

Seules les règles de production vérifiant le format Bissectrice $\rightarrow$ t' seront déclenchées si le premier point de t' appartient à la zone AngleZone. Une fois une production déclenchée, elle sera testée. Le bloc de préconditions est valide si tous les CSD nécessaires à la production sont valides. Si les préconditions sont validées, les contraintes seront testées. Le fait qu'elles soient satisfaites entraînera la réduction de la production et la création de nouveaux CSD en postcondition. Ce processus, comme pour celui des GMC, est répété jusqu'à stabilité. Le résultat de l'analyse d'un tracé consiste donc en une (ou plusieurs) séquence de réductions.

4.3.2 Évaluation d'une production

Plusieurs interprétations étant possibles pour un seul tracé, une prise de décision est nécessaire pour choisir la plus adéquate. Pour ce faire, la notion de logique floue est introduite dans le formalisme GMC-PC. **À chaque règle de production est associé un degré d'adéquation** $\rho \in$ **[0, 1]**. Ce degré dépend de la satisfaction des blocs de préconditions et contraintes.

Évaluation des préconditions

Cette évaluation se base sur la notion de positionnement relatif flou [Blo99]. À chaque zone du document modélisée par le terme *[position]* dans un CSD, est associée une fonction d'appartenance dénotée $u^{\gamma}{}_{\text{position}}$. Cette fonction, appelée *paysage flou*, correspond au degré d'appartenance d'un élément E à un contexte structurel γ [position]. La figure 4.4 illustre un exemple de paysage flou modélisant l'appartenance à un rectangle, telle que le degré d'appartenance dépend de la luminosité.

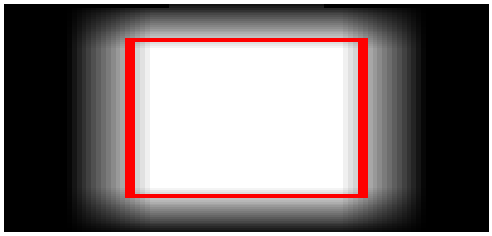


FIGURE 4.4 – Paysage flou modélisant la relation "rectangle[Dans]" [MA09]

Le calcul du degré d'appartenance dépend de la *partie* de l'élément E qui est considérée. Par exemple, sur la figure 4.5, deux fonctions d'appartenance sont représentées. Sur la figure de gauche (CSD : γ [position] tracé [un]), seul un point est considéré. On considère ici l'opérateur *un*, *i.e.* le degré d'appartenance du tracé E au paysage flou correspond au degré d'appartenance du point x $\in$ E le plus proche du paysage flou :

$$u^{\gamma}{}_{\text{position}}(\text{E}) = \max_E \; u^{\gamma}{}_{\text{position}}(\text{x}).$$

Si l'on considère l'opérateur *tous* tel que sur la figure de droite (CSD : γ [position] tracé [tous]), le degré d'appartenance de E correspond à la valeur moyenne de tous les degrés d'appartenance de ses points :

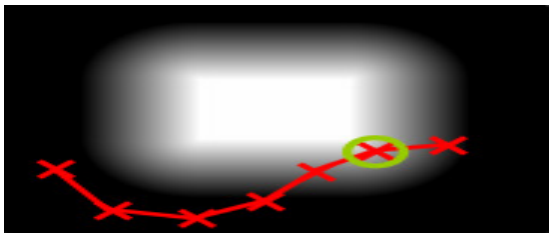
$$u^{\gamma}{}_{\text{position}}(\text{E}) = \frac{1}{|E|} \sum_E u^{\gamma}{}_{\text{position}}(\text{x}).$$

Pour calculer le degré d'adéquation des préconditions $p^{\text{préconditions}}$, les degrés d'adéquation des différents CSD sont fusionnés par un opérateur de fusion T-norme.

Évaluation des contraintes

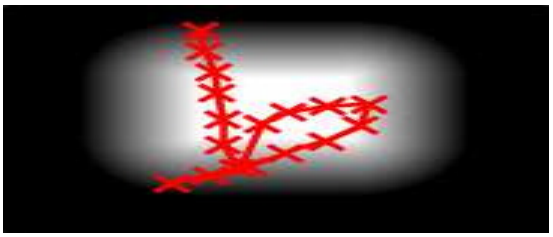
Les systèmes de reconnaissance utilisés dans [MA09] se basent sur des systèmes d'inférence floue [TS85], permettant d'obtenir des scores de confiance entre 0 et 1 relatifs à la reconnaissance statistique de la forme analysée.

CSD : γ [position] tracé [un]



$u^{\gamma}{}_{\text{position}}(\text{E}) = \max_E u^{\gamma}{}_{\text{position}}(\text{x})$

CSD : γ [position] tracé [tous]



$u^{\gamma}{}_{\text{position}}(\text{E}) = \frac{1}{|E|} \sum_E u^{\gamma}{}_{\text{position}}(\text{x})$

FIGURE 4.5 – Exemples de degrés d'appartenance pour [un] et [tous] [MA09]

Le score d'adéquation d'une production p est enfin déduit de ces deux scores cités plus haut et est de la forme suivante :

$$\rho_p = p^{\text{contraintes}} \cdot p^{\text{préconditions}} \tag{4.1}$$

Il en découle que le score d'une interprétation, qui est une séquence de réduction de règles de productions, est la combinaison des scores d'adéquation de toutes les règles réduites par la séquence. L'analyseur choisit donc l'interprétation ayant le score le plus élevé.

La notion de *rejet* (d'absence d'interprétation valide) est aussi présente, sous deux formes :

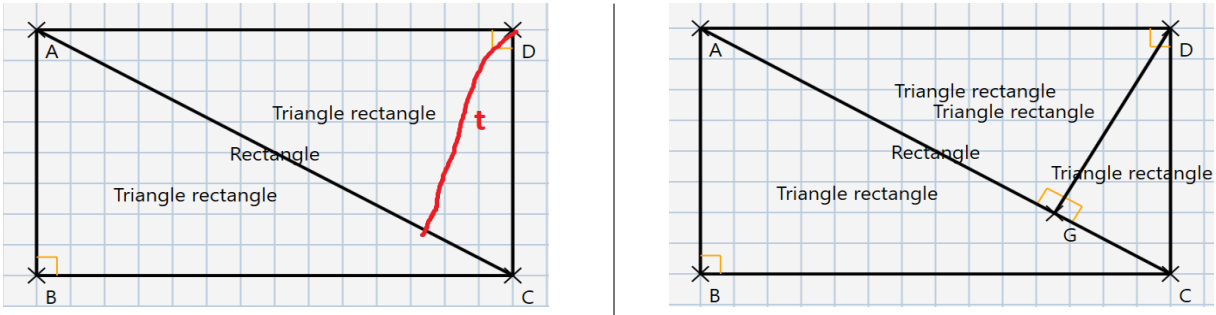
— *Rejet de distance* : toutes les interprétations ont un score moins élevé qu'un seuil d'acceptabilité défini par le concepteur ;

— *Rejet d'ambiguïté* : la différence de score entre la meilleure interprétation et sa dauphine est inférieure à un certain seuil, dans ce cas, l'utilisateur peut être sollicité pour choisir l'interprétation adéquate.

4.3.3 Construction de l'arbre d'analyse et stratégies de recherche

L'analyse d'un nouveau tracé t, tel que celui présenté dans la figure 4.6, conduit à la construction d'un **arbre d'analyse,** dont une partie est illustrée dans la figure 4.7. La racine de l'arbre est le tracé, et les noeuds et les feuilles représentent les règles déclenchées. Le chemin en bleu pointillé représente le résultat de l'analyse, *i.e.* la séquence des règles de production réduites pour trouver la bonne interprétation.

En suivant cet exemple, on peut résumer l'interprétation comme suit : le tracé t va être



(a) Tracé dessiné

(b) Résultat de l'analyse

FIGURE 4.6 – Processus d'analyse du tracé t

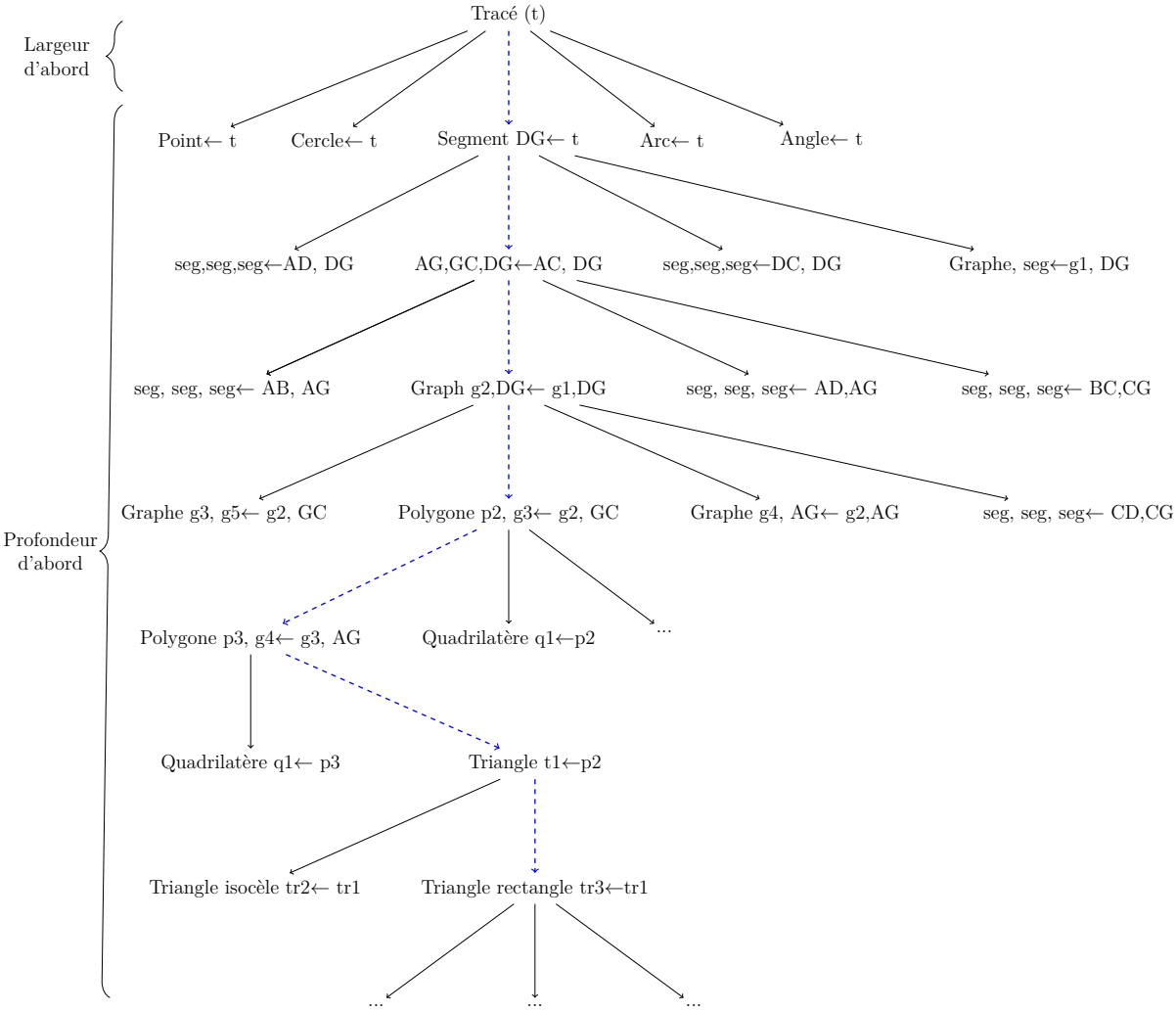


FIGURE 4.7 – Partie de l'arbre d'analyse

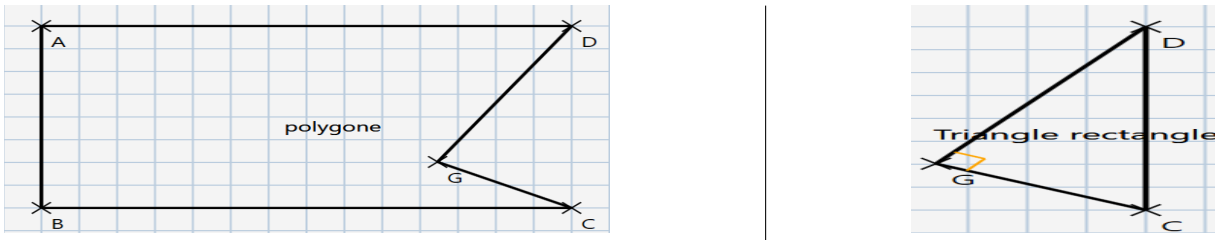
transformé d'abord en un segment DG, qui va lui même diviser le segment AC en trois nouveaux segments AG, CG, et DG (dans l'arbre d'analyse, cette division est illustrée par la règle : *AG, GC, DC* ← *AC, DG*). Cette division va engendrer la mise à jour de la structure du document, et notamment les connexions entre segments, représentés dans un graphe de connexion (lui même élément à part entière de la grammaire, appartenant à l'ensemble V_N, au même titre que les cercles). La règle *Graphe g2,DG*← *g1,DG* illustre cette mise à jour, g1 étant le graphe représentant l'état des segments connectés avant la division. La division des segments mettant à jour la structure, le graphe g1 est remplacé par le nouveau graphe g2. Les zones de contact de ces graphes recouvrent les extrémités des segments connectés, ce qui permet de déclencher la règle de création de polygone lorsqu'un nouveau segment intersecte deux zones de contact du graphe, créant ainsi un cycle. La règle *Polygone p2, g3*← *g2, GC* illustre la création d'un nouveau polygone. À partir de là, cette forme peut être transformée en n'importe quelle forme géométrique (octogone, pentagone, quadrilatère, etc), et donc ces règles sémantiques seront déclenchées jusqu'à trouver la bonne interprétation. Nous développerons plus en détail les productions de polygones et de graphes dans le chapitre 5, où nous présenterons l'adaptation du formalisme GMC-PC au domaine de la géométrie. La grammaire offre aussi la possibilité de spécifier le type d'exploration (*largeur-d'abord* ou *profondeur-d'abord*). Le type d'exploration est assigné à chaque règle en amont. Par défaut, la stratégie d'exploration est en largeur d'abord. Pour l'exploration en profondeur-d'abord, l'analyseur va réduire la première règle applicable et stopper la recherche. Les éléments résultants et les CSD (de la réduction) vont être les points de départs de la relance de l'analyse. Pour la stratégie largeur-d'abord, l'analyseur explore toutes les interprétations possibles et choisit l'hypothèse qui a le meilleur score. La figure 4.7 illustre le fait que pour les productions qui constituent un risque d'ambiguïté, tel qu'un segment ou un arc, on adopte une stratégie d'exploration en largeur-d'abord. Pour les autres productions (division de segments, création de polygones par exemple), on adopte une stratégie d'exploration en profondeur-d'abord. Au-delà, de ce choix entre stratégies conduit par le risque d'ambiguïté d'une production, on peut remarquer qu'une hiérarchie de concepts est naturellement établie entre les productions (e.g un triangle peut-être réduit en un triangle rectangle).

4.4 Limites de GMC-PC et extension du formalisme

Dans cette section nous présentons les limites de GMC-PC que nous avons identifiées, et présentons succinctement nos propositions pour pallier à celles-ci. Nous pouvons voir que pour un exemple relativement simple (*cf.* Figure 4.7), le processus d'analyse est assez fastidieux. Au fur et à mesure que le document se complexifie, la combinatoire devient impossible à gérer pour l'analyseur, et la contrainte d'interprétation temps-réel des tracés de l'utilisateur n'est plus respectée. Ceci est du à plusieurs facteurs. Il est vrai que le pilotage de l'analyse permet de contrôler la combinatoire, en faisant en sorte que seules les règles pertinentes soient déclenchées (au lieu de tester toutes les combinaisons possibles). Cependant, il est aussi clair dans l'arbre d'analyse de la figure 4.7) que ce pilotage par le contexte n'est pas capable de limiter le déclenchement de règles de division de segments par exemple, alors que la seule règle de division pertinente a déjà été réduite (la règle *AG, GC, DG* $\leftarrow$ *AC, DG* dans le troisième niveau/itération de l'analyse, en supposant que le premier niveau soit *Tracé(t)*). On voit bien que ces règles continuent à être déclenchées au niveau plus bas (*e.g.* la règle *seg, seg, seg* $\leftarrow$ *AB, AG*). Cela ajoute une complexité artificielle au processus d'analyse. Ce problème est commun à toutes les règles et vient du fait que l'analyseur déclenche toutes les règles contextuellement cohérentes. Pour parer à ce problème, **nous proposons d'introduire la notion de hiérarchie dans les règles de production**, qui va permettre de structurer l'arbre d'analyse en couches, qui correspondent aux niveaux d'interprétation du domaine d'application. Par exemple, nous pouvons dégager de l'arbre d'analyse précédent une première couche de reconnaissance de primitives graphiques, une seconde de mise à jour de la structure du document (intersection d'éléments, divisions, fusion, etc), et des couches sémantiques (une couche pour la gestion des connexions par la création de graphes sémantiques, une couche pour la création de polygones par la détection de cycles dans ces graphes, et une couche sémantique pour détecter des figures géométriques à partir des polygones générés). Dans GMC-PC, il n'est possible de définir que deux niveaux d'interprétations qui correspondent aux deux stratégies possibles de recherche : largeur-d'abord et profondeur-d'abord. Notre extension de la définition du formalisme définira un processus d'analyse à la volée tant **guidé par le contexte que par la hiérarchie**. Cette proposition de grammaire étendue que nous appellerons GMC-HPC (pour *Grammaires Multi-ensembles à Contraintes Hiérarchique et Guidée par le Contexte*), sera détaillée dans le chapitre 5.

Une autre limite est la complexité générée par la création de polygones. Dans l'exemple

de la figure 4.6, le nouveau segment [GC] va générer deux nouveaux cycles dans le graphe de connexion g2 et donc avoir comme conséquence la création de trois polygones illustrés dans la figure 4.8 : le triangle DGC, et le polygone ADGCB. La règle de création d'un



(a) ADGCB (b) DGC

FIGURE 4.8 – Deux polygones créés correspondant aux deux cycles générés par [GC]

polygone *Polygone p2, g3← g2, GC* (niveau 5) va donc devoir être consommée deux fois pour générer ces figures. Un processus similaire est induit par la création du segment [AG]. Cela aussi créé de la complexité artificielle qui ralentit le processus d'analyse, puisqu'à chaque nouvelle itération/niveau, l'analyseur déclenche toutes les règles contextuellement valides. Dans l'idéal, la génération de ces deux polygones devrait se faire en consommant la règle une unique fois.

La tâche de la modélisation de la connaissance par les règles de production revient au concepteur, ces règles étant de la forme $\alpha \leftarrow \beta$, tels que α et β sont des multi-ensembles d'éléments. Le nombre d'éléments β étant fixé dans la partie droite de la règle, il n'est pas possible d'exprimer la création de n polygones à partir d'un segment et d'un graphe sémantique de connexion. Nous estimons que, au delà du contexte local du tracé, et du contexte structurel modélisé par les préconditions et postconditions, il existe un **contexte global sémantique** qui peut être modélisé par des graphes sémantiques qui représentent les connexions entre les éléments. **Pour alléger la complexité de l'analyse, nous proposons de modéliser ce contexte global par un analyseur contextuel sémantique** qui sera à même de détecter des patterns (formes) spécifiques au domaine d'application (à l'instar des cycles dans un graphe). Nous externalisons donc cette étape de l'analyse, ce qui aura comme impact d'éviter l'explosion de la combinatoire. Ces propositions seront détaillées dans le chapitre 5 suivant.

Chapitre 5

Grammaire Multi-ensembles à Contraintes Hiérarchique et Guidée par le Contexte

Dans le chapitre précédent, nous avons présenté le formalisme de base des GMC-PC, son expressivité et ses limites. Nous détaillons dans ce chapitre notre approche pour pallier ces limites, en vue de respecter la contrainte d'interaction temps-réel avec l'utilisateur lorsque l'on va s'intéresser à la composition de documents complexes. Dans le contexte de la reconnaissance de figures géométriques, l'extension du formalisme que nous proposons permettra la supervision de la stratégie de l'élève et la génération immédiate de feedback personnalisés (feedback de guidage et de correction). Notre approche se décline en deux stratégies. *La première stratégie* consiste à étendre la définition de GMC-PC vers une grammaire hiérarchique GMC-HPC. Pour ce faire, nous introduisons la notion de couches logiques dans la définition des règles, ce qui va permettre au concepteur de définir autant de niveaux d'interprétations que nécessaire selon le domaine d'application spécifique qu'il modélise. Nous introduisons aussi la notion de hiérarchisation des contextes, qui va permettre d'accélérer le processus de vérification de la cohérence des CSD (Contextes Structurels du Document) dans le bloc de préconditions. *La deuxième stratégie* consiste à externaliser la recherche de contexte structurel global, par la création d'un analyseur contextuel sémantique, qui sera en charge d'interpréter des structures complexes, souvent à composition récursive, d'y détecter des patterns particuliers (*e.g.* des cycles), et de transformer ces patterns en figures structurées complexes (typiquement des polygones). Ces figures seront ensuite redonnées en point d'entrée à la reprise du processus d'analyse associée au formalisme GMC-HPC. Cela permettra d'éviter la complexité artificielle générée par l'application d'une même règle n fois pour générer n polygones, par exemple.

Nous avons, au début de notre travail sur cet axe de la thèse, essayé d'adapter la grammaire de base au domaine de la géométrie par la définition des règles de production

relatives au domaine. Par la suite, en nous heurtant au problème de la complexité de l'analyse, nous avons proposé les stratégies introduites plus haut. Ces propositions se veulent génériques, pour démontrer cette propriété, nous avons aussi adapté la grammaire pour le domaine de la composition de plans d'architecture. Nous commencerons donc ce chapitre par une vue globale de la modélisation de la connaissance à priori avant d'attaquer notre extension du formalisme.

5.1 Modélisation de la connaissance géométrique a priori

Notre première contribution a été de modéliser l'ensemble des différents objets géométriques enseignés au collège, que l'on peut répartir en deux catégories, du point de vue reconnaissance de formes :

— les primitives graphiques : point, segment, droite, cercle, arc, angle ;
— les figures géométriques : tous les types de polygones, les triangles, quadrilatères, pentagones...

Nous parcourons dans cette section notre approche pour représenter la connaissance du domaine dans les règles de production, avec l'encapsulation des théorèmes de géométrie dans le bloc de contraintes. Nous avons vu l'exemple de la production d'un angle en GMC-PC dans la section 4.2 (figure 4.1). Nous nous intéresserons ici plus particulièrement aux productions définissant les figures géométriques structurées complexes. Pour pouvoir définir un polygone dans la grammaire, il est nécessaire de maintenir les relations de connexions et de dépendance entre les différents segments présents dans le document. Nous avons mis en place pour ce faire sur une structure intermédiaire, appartenant à l'ensemble V_N de la grammaire, que nous avons dénommée *graphe sémantique de connexion*. Ces graphes sont caractérisés par une *structure récursive*, puisqu'on peut entièrement définir la connaissance relative à ce type d'éléments en traduisant les deux principes suivants en règles de production :

1. Deux symboles connectés dans le document forment un nouveau graphe sémantique ;
2. Un symbole et un graphe connectés forment un nouveau graphe.

Dans la sous-section suivante, nous allons voir en détail les règles de production relatives à l'interprétation des figures complexes en géométrie (les polygones) à partir de

la notion de graphe sémantique de connexion. Reconnaître ces polygones consiste donc à retrouver des cycles dans les graphes. Ces cycles représentent les patterns d'intérêt particuliers non seulement pour la composition de figures géométriques, mais aussi pour les plans d'architecture pour détecter les pièces manuscrites. Notons que cette modélisation est générique et applicable à plusieurs types de documents structurés (l'interprétation des molécules pour la composition de schémas en chimie, l'interprétation des circuits électriques, etc).

5.1.1 Graphe sémantique de connexion et polygones

Vu l'infinité de possibilités inhérentes à la création de polygones de n cotés, et donc l'impossibilité de modéliser cette connaissance à priori par des règles de format

$$Polygone\ p \leftarrow segment_1\ segment_2, ...segment_n,$$

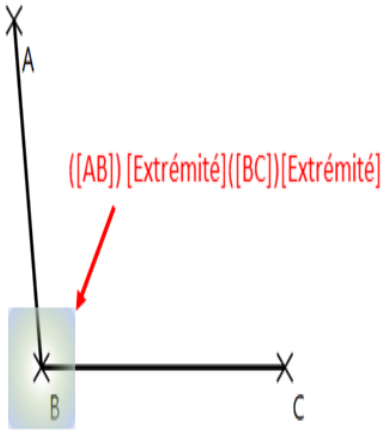
pour tout n, il est nécessaire de modéliser, pour chaque sous-ensemble d'éléments connectés, un graphe sémantique de connexion.

L'élément graphe a la particularité de ne pas être visible à l'utilisateur, et de ne pas modifier la structure du document. Il est cependant nécessaire au maintien des liens de connexions entre les éléments, en vue de créer des polygones. Les noeuds de ce graphe représentent les extrémités des segments qui y appartiennent, tandis que les arcs modélisent les relations de connexion. La création de polygone est modélisée par un segment qui serait connecté à deux noeuds de ce graphe, cette création est modélisée par une règle de production de la forme

Polygone p, Graphe g2, segment s1 ← Segment s1, Graphe g1.

Cette modélisation signifie que le segment s1 ne sera pas consommé par la règle, tandis que le graphe g1 est transformé en graphe g2, la différence entre g1 et g2 étant la relation de connexion des deux extrémités de s1 qui ont contribué à créer un cycle. La définition des graphes de connexion est gérée par deux règles de production. La première, illustrée dans la figure 5.1, est relative à la création d'un graphe à partir de deux segments adjacents, n'appartenant pas par ailleurs à des graphes de connexion existants (*cf.* bloc de contraintes dans 5.1.a). La deuxième, illustrée dans la figure 5.2, est relative à un sous-ensemble de segments connectés déjà représentés par un graphe, et un nouveau segment, qui n'appartient pas à ce graphe (g1 et [FN] dans la figure 5.2.b). Le CSD en précondition

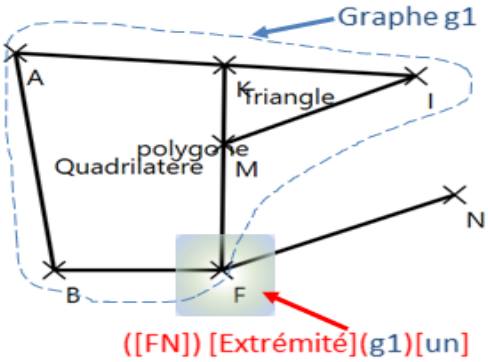
Graphe : g, s1, s2 → segment s1, segment s2
avec :
Préconditions :
(s1) [Extrémité] (s2) [extrémité]
Contraintes :
Adjacents(s1, s2)
NappartientPasAGraphe(s1)
NappartientPasAGraphe(s2)
Postconditions :



a) Règle de production b) Composition

FIGURE 5.1 – Création de graphe à partir de deux segments

Graphe : g2, s1 → segment s1, Graphe g1
avec :
Préconditions :
(s1) [Extrémité] (g1) [un]
Contraintes :
NappartientPasAGraphe(s1, g1)
Postconditions :



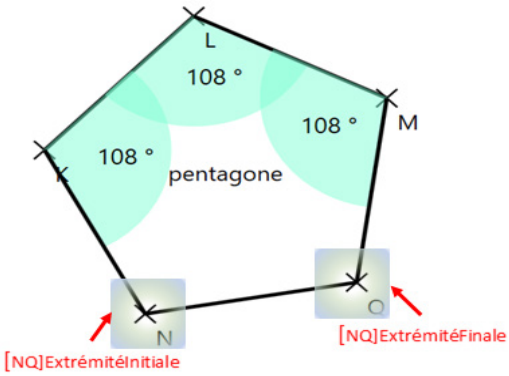
a) Règle de production b) Composition

FIGURE 5.2 – Mise à jour des connexion dans GMC-PC

dans la figure 5.2.a) modélise le fait qu'[un] des noeuds du graphe g1 appartient à la zone extrémité du segment [FN].

Cette modélisation des graphes nous permet d'appréhender la création de polygone, illustrée dans la figure 5.3. Pour composer un pentagone (*cf.* figure 5.3.b, il faut que le nouveau segment [NQ] soit connecté au graphe g1 par ses deux extrémités. Le polygone p1 est réduit, et à partir de ce dernier, la règle de création de pentagone va être testée, parmi toutes les autres règles de polygones particuliers. La production d'un pentagone, illustrée dans la figure 5.4, se base sur la définition mathématique d'un pentagone (5 cotés, et tous les angles de valeur 108 °), qui est encapsulée dans le bloc de contraintes.

Polygone p1 : Graphe g2, s1 → segment s1, Graphe g1 avec :
Préconditions :
(s1) [ExtrémitéInitiale] (g1) [un] &
(s1)[ExtrémitéFinale] (g1) [un]
Contraintes :
NappartientPasAGraphe(s1, g1)
Postconditions :



a) Règle de production b) Composition

FIGURE 5.3 – Création d'un polygone dans GMC-PC

Pentagone : pent → Polygone p avec :
Préconditions :
Contraintes :
EstPentagone(p) //vérification que le polygone a 5 cotés égaux et que tous les angles intérieurs sont égaux à 108°
Postconditions :
(pent)[Dans] (Cercle : c) [centre] $\Longrightarrow$ [Cercle-Circonscrit → c]

FIGURE 5.4 – Règle de production d'un pentagone en GMC-PC

5.1.2 Production de segments

La figure 5.5 représente la production d'un segment dans GMC-PC. Que ce soit dans le domaine de la géométrie, ou la composition des plans architecturaux (murs), c'est cet élément qui constitue la base de l'information structurelle du document (un segment va contenir des extrémités -points-, et la connexion entre segments va constituer tous les types de figures géométriques). Il est donc aussi à la base de la complexité du processus d'analyse, et plusieurs règles sont mises en oeuvre afin de gérer les différentes interactions entre ces segments. Nous avons déjà vu certaines de ces interactions dans la section 4.3, la division de deux segments en trois nouveaux, la gestion des connexions en utilisant des graphes sémantiques de connexion, et la règle création de polygones. Tous les CSD relatifs au déclenchement de ces règles sont présents dans le bloc de postconditions de la règle de production d'un segment.

Le bloc de préconditions illustre la fait de pouvoir avoir plusieurs contextes cohérents possibles pour la création d'un segment. Ce bloc est composé d'une disjonction de quatre préconditions, qui modélisent le fait qu'un tracé t pouvant être réduit en un segment *res* peut appartenir à plusieurs contextes possibles :

1. Contexte ⓐ : ses deux points d'extrémité (premier & dernier) appartiennent à une zone d'extrémité de deux symboles existants S1, S2 (ⓐ, figure 5.6.a) ;
2. Contexte ⓑ ou ⓒ : un de ses points d'extrémité (premier ou dernier) appartient à une zone d'extrémité d'un symbole existant S1 ;
3. Contexte ⓓ : aucun de ses points n'appartient à une zone d'extrémité d'un symbole existant.

Quand il existe plusieurs contextes possibles pour une règle de production, l'espace de recherche contextuel exploré par l'analyseur est similaire à l'espace de recherche des règles. Et donc le processus d'analyse est similaire. L'analyseur met en compétition ces différentes hypothèses contextuelles de création de segment, ainsi pour trois hypothèses valides, il y a création de trois branches dans l'arbre d'analyse, chacune relative à l'une des interprétations possibles de la règle. Pour ces trois branches, le processus d'analyse se déroule comme décrit dans la section 4.3 , pour ensuite choisir le contexte (la précondition) qui induit la meilleure interprétation possible -*i.e.* qui a le meilleur score-. On en déduit donc, que même si le pilotage par le contexte introduit dans le formalisme GMC-PC permet d'alléger la complexité de l'analyse par rapport à formalisme classique GMC, la possibilité d'avoir plusieurs contextes dans le même bloc de préconditions peut complexi-

Segment : res → tracé t avec : *type d'exploration = 'largeur'*
Préconditions :
FirstPrecondition
(ⓐ (Symbole :S1) [Extrémité] (t) [premier] **&** (Symbole :S2) [Extrémité] (t) [dernier]**)**
Ou ⓑ (Symbole : S1) [Extrémité] (t) [premier]
Ou ⓒ (Symbole : S1) [Extrémité] (t) [dernier]
Ou ⓓ (Document) [in] (t) [all]
Contraintes :
Reconnaisseur(t, Segment)
Postconditions :
(res)[LongueurSegment] (tracé : t) [extrémité] $\Longrightarrow$ [angle → t] //si l'extrémité d'un tracé t appartient à la zone longueurSegment la règle de création d'angle est déclenchée par l'analyseur

(res)[LongueurSegment] (Segment : s1) [extrémité] $\Longrightarrow$ [segment, segment, segment → res, s1] //si une extrémité d'un segment s1 intersecte la zone LongueurSegment de res, déclenchement de la règle de division des deux segments res et s1 en trois nouveaux segments

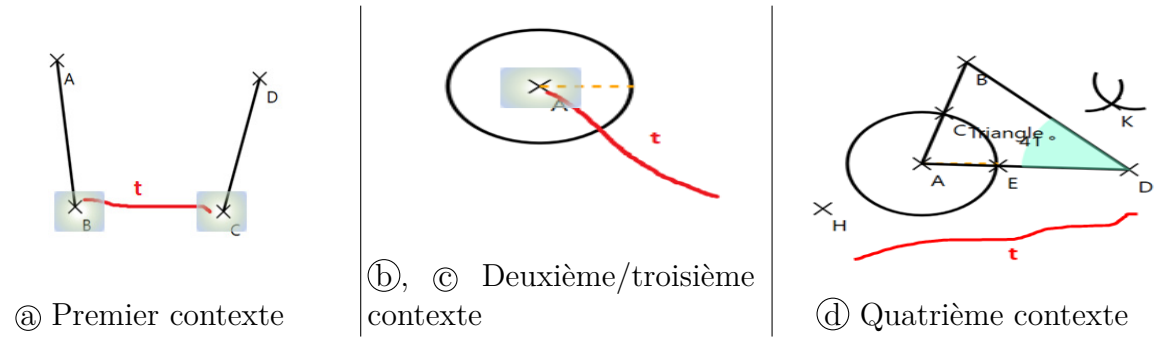
(res)[LongueurSegment] (Segment : s1) [un] $\Longrightarrow$ [segment, segment, segment, segment → res, s1] //si un point d'un segment s1 intersecte la zone LongueurSegment de res, déclenchement de la règle de division des deux segments res et s1 en quatre nouveaux segments

(res)[Extrémité] (Segment : s1) [extrémité] $\Longrightarrow$ [Graphe, res, s1 → res, s1] //si une extrémité d'un segment s1 appartient à une zone d'extrémité de res, déclenchement de la première règle de création d'un graphe

(res)[Extrémité](Graphe : g)[extrémité] $\Longrightarrow$ [Graphe, res → g, res] //si une extrémité d'un graphe g appartient à une zone d'extrémité de res, déclenchement de la deuxième règle de création d'un graphe

(res)[premièreExtremité](Graphe : g)[extrémité] &
(res)[derniéreExtremité](Graphe : g)[extrémité] $\Longrightarrow$ [Polygone, Graphe, res → g, res] //si une extrémité d'un graphe g appartient à une zone d'extrémité de res, déclenchement de la deuxième règle de création d'un graphe

FIGURE 5.5 – Règle de production d'un segment en GMC-PC



ⓐ Premier contexte

ⓑ, ⓒ Deuxième/troisième contexte

ⓓ Quatrième contexte

FIGURE 5.6 – Différents contextes pour l'interprétation d'un tracé t en segment de la figure 5.5

fier le processus d'analyse. Dans [PMA10], les auteurs essaient de résoudre ce problème en introduisant la notion de **priorisation des préconditions**, représentée dans la figure 5.5 par le marqueur **FirstPrecondition**. Le principe de ce marqueur est d'introduire la possibilité de choisir la première précondition valide, sans la mettre en compétition avec les préconditions suivantes. C'est donc le même principe que la recherche de production par la stratégie en profondeur d'abord qui permet d'accélérer l'analyse en choisissant la première production valide. Dans l'exemple de la figure 5.6.a, c'est la première précondition qui va donc être prise en compte, par conséquent, le segment résultant *res* sera remis au propre en liant ses extrémités aux deux segments existants [AB] et [CD]. La décision d'appliquer ce marqueur *FirstPrecondition* revient au concepteur, pour stopper la mise en concurrence des préconditions, quand le risque de confusion semble *a priori* limité.

5.2 Hiérarchisation des contextes

Au delà des stratégies d'exploration des préconditions cohérentes, il existe aussi différents contextes possibles au sein d'une même précondition. La figure 5.7 illustre ce fait. Avec l'apport de l'opérateur FirstPrecondition, le formalisme valide la première précondition de la règle (ⓐ dans la figure 5.5), qui stipule que *t* est lié à deux symboles existants *s1* et *s2*. Dans cet exemple, l'espace de recherche au sein de la précondition est le couple de symboles (S1, S2) qui sont liés au tracé *t*. La limite du formalisme existant, enrichi par l'opérateur Firstrécondition, est qu'il n'y a pas de stratégie d'exploration alternative dans cette seconde couche de la recherche de contexte. En effet, au sein de cette précondition, il existe, pour l'exemple de la figure 5.7 six hypothèses contextuellement valides : *t* peut être lié aux couples (AB, EF), (AB, BE), (AB, EC), (AD, BE), (AD, EF) or (AD,

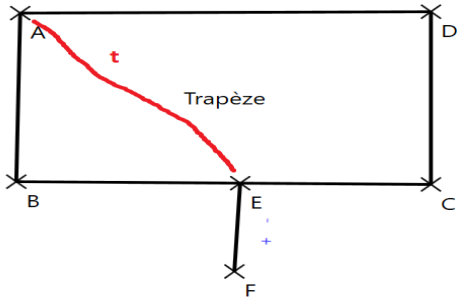


FIGURE 5.7 – Limite de la priorisation des préconditions

EC). Puisqu'il n'existe qu'une seule stratégie d'exploration (largeur d'abord), l'analyseur considère ces hypothèses en tant que règles de production de segment indépendantes. Par conséquent, six branches équivalentes seront créés dans l'arbre d'analyse (*cf.* figure 5.8), et donc la complexité de l'analyse sera artificiellement multipliée par six.
Afin d'établir des stratégies cohérentes sur tous les nivaux de l'analyse (au niveau des règles, au niveau des préconditions, au niveau des contextes), **notre première extension du formalisme GMC-PC consiste à définir un nouvel opérateur *FirstContext*.** Cet opérateur va permettre d'établir une stratégie d'exploration en profondeur-d'abord au niveau de la recherche de contextes cohérents au sein d'une précondition. La combinaison de *FirstPrecondition* et **FirstContext** force donc l'analyseur à stopper la recherche de contexte à la première hypothèse contextuelle valide au sein de la première précondition valide (*i.e.* le couple (AB, EF) dans cet exemple). L'impact sur la complexité de l'analyse est illustré pour l'exemple *via* l'arbre d'analyse dans la figure 5.10 ; on peut y voir que ce nouvel opérateur va permettre de drastiquement réduire cette complexité, sans pour autant perdre d'information sur les connexions, dans les nombreux cas où évter la mise en concurrence des contextes est un risque raisonnable.

La figure 5.7 illustre le cas où le tracé est relié par sa première extrémité à un symbole existant. Une limite de l'opérateur *FirstContext* se présente lorsque deux zones d'extrémités se chevauchent, et que le premier point du tracé t appartient à la zone de chevauchement. Par exemple, la figure 5.11 illustre le cas où le tracé est relié par sa première extrémité à un symbole existant (précondition ⓑ dans la figure 5.9), et où les deux segments [AB] et [CD] ont des zones d'extrémités qui se chevauchent. Dans ce cas précis, la mise en compétition des deux contextes *[AB][Extremite](t)[Premier]* et *[CD][Extremite](t)[Premier]* pourrait être pertinente. Le calcul du degré d'appartenance de t aux deux zones permet de trouver la meilleure interprétation (celle ayant le meilleur

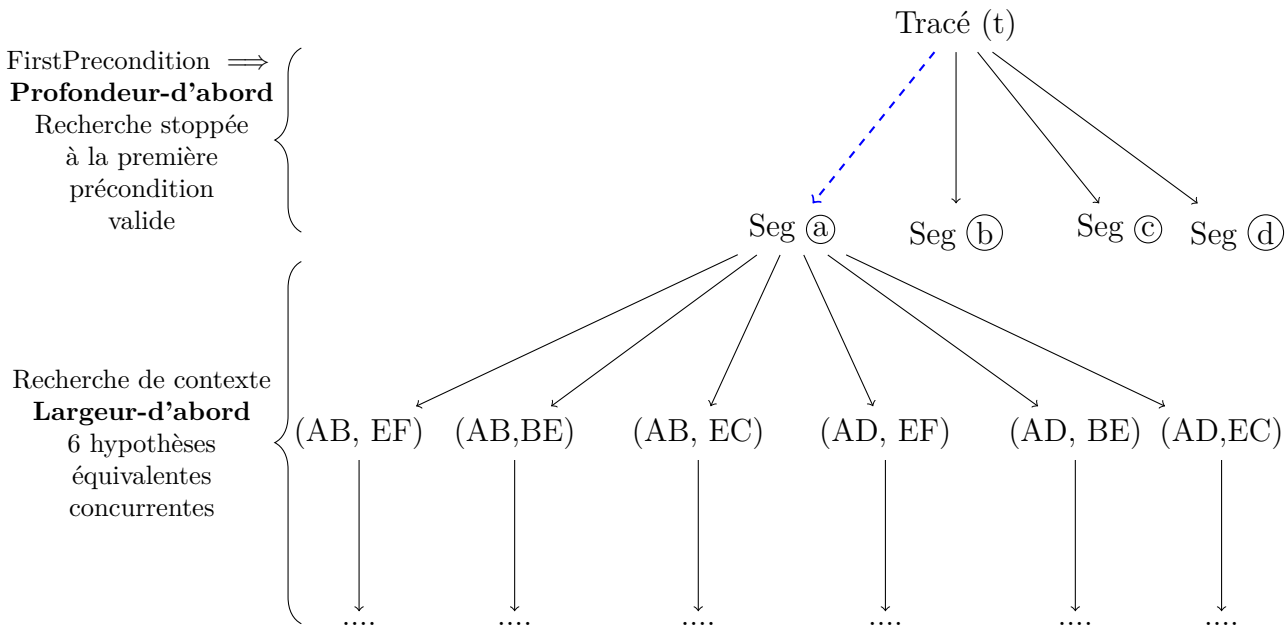


FIGURE 5.8 – Analyse avec FirstPrecondition

Segment : res → **tracé t** avec :
Preconditions :
FirstPrecondition
FirstContext
(ⓐ (Symbole :S1) [Extrémité] (t) [premier] **&** (Symbole :S2) [Extrémité] (t) [dernier])
Ou ⓑ (Symbole : S1) [Extrémité] (t) [premier]
Ou ⓒ (Symbole : S1) [Extrémité] (t) [dernier]
Ou ⓓ (Document) [in] (t) [all]

FIGURE 5.9 – Introduction de l'opérateur FirstContext dans les préconditions

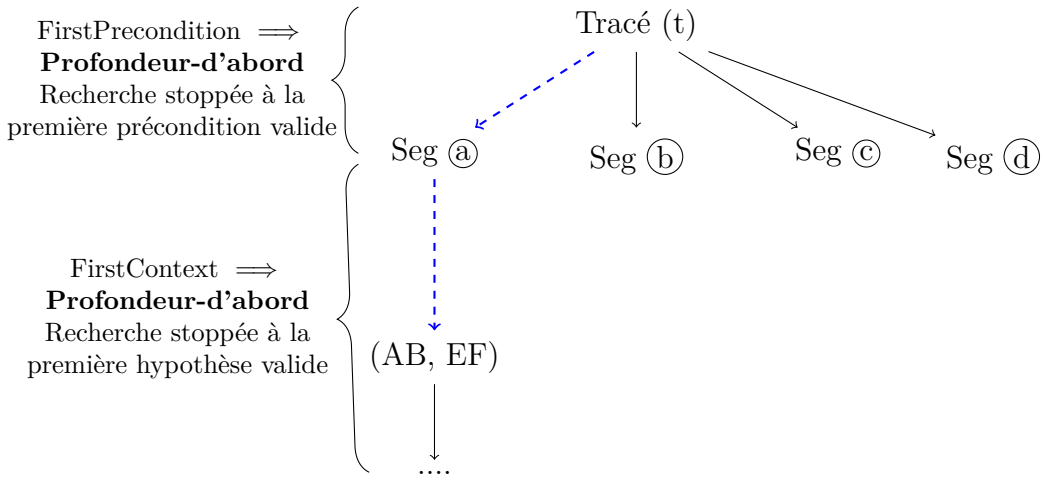


FIGURE 5.10 – Analyse avec FirstPrecondition et FirstContext

degré d'appartenance). L'ajout de l'opérateur *FirstContext* entraîne une stratégie de recherche en profondeur d'abord et contourne donc cette mise en compétition (à tort ici) : *FirstContext* force l'analyseur à choisir la première interprétation valide, qui, dans cet exemple, n'est pas forcément la meilleure. Cependant, la robustesse de cette extension réside dans le fait de pouvoir interagir avec l'utilisateur dans le cadre de l'interprétation à la volée de ses tracés manuscrits. En effet, il a la possibilité de valider implicitement l'interprétation en continuant sa production, ou bien de réagir explicitement en supprimant le segment reconnu par le système et en le re-dessinant de manière plus précise. Nous avons donc fait le choix d'un compromis entre complexité de l'analyse et précision de l'interprétation, pour les règles où nous acceptons de prendre ce risque et par conséquent de solliciter l'utilisateur pour qu'il re-dessine son tracé.

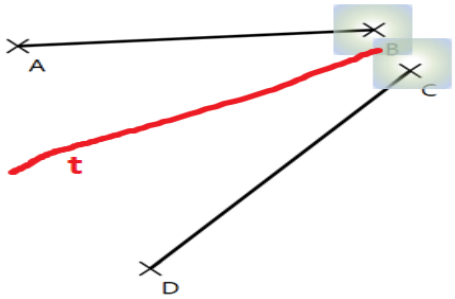


FIGURE 5.11 – Exemple de cas où la recherche en profondeur n'aboutit pas à la bonne interprétation

5.3 Hiérarchisation des règles de production

Nous avons vu dans la section 4.3 la possibilité pour le concepteur de formaliser deux stratégies d'exploration dans le processus d'analyse associé au formalisme GMC-PC, largeur-d'abord et profondeur-d'abord. Cela se traduit dans la formalisation des règles de production, comme par exemple dans règle illustrée dans la figure 5.5, où l'on voit le type d'exploration fixé à *largeur* (**Segment : res** $\rightarrow$ **tracé t** avec : *type d'exploration = 'largeur'*). Comme évoqué dans la section 4.4, avoir uniquement deux stratégies d'exploration constitue une limite dans le domaine de la géométrie. En effet, nous remarquons que la majorité des règles peut être assignée à une stratégie d'exploration en profondeur, mais cela n'empêche pas l'explosion combinatoire. Cela s'explique par le fait qu'il peut y avoir un grand nombre de règles déclenchées par le contexte qui doivent être évaluées avant de trouver une règle applicable.
Nous constatons aussi qu'il existe plusieurs couches logiques d'interprétation d'un document structuré, en particulier pour le domaine de la géométrie. En effet, nous pouvons distinguer quatre niveaux/couches d'interprétation :

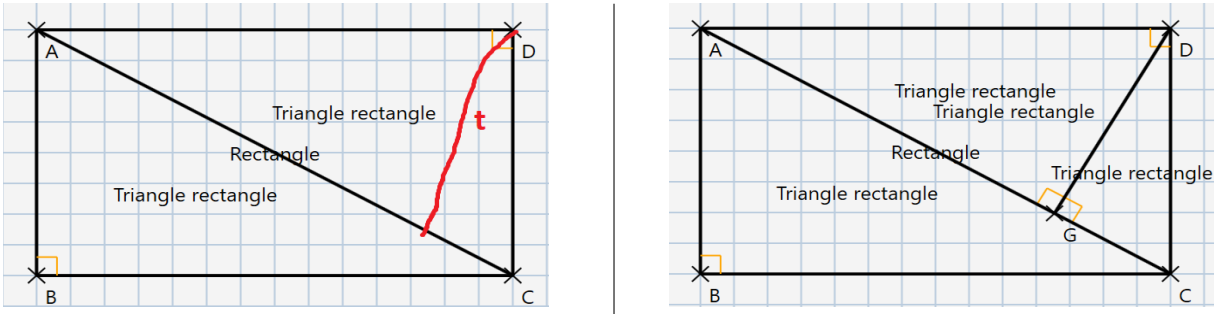
— une première couche structurelle d'interprétation qui correspond à la génération de primitives graphiques (point, segment, arc..) à partir d'un tracé. Comme souligné dans la section 4.3, c'est à ce niveau qu'il y a un risque d'ambiguïté dans l'interprétation, et donc les règles sont associées à une exploration de type largeur d'abord ;
— une deuxième couche structurelle qui correspond aux règles modélisant l'interaction structurelle entre les éléments du documents (*e.g.* intersection arc-arc, intersection arc-segment, division/fusion de segments...).
— La troisième couche sémantique qui correspond au maintien et à la mise à jour des liens de connexion et de dépendance entre les différents éléments ainsi que la détection de patterns particuliers (*e.g.* graphe de connexion et polygone vus dans la section 5.1) ;
— Une dernière couche sémantique qui consiste à transformer les polygones reconnus en des figures géométriques, où les définitions mathématiques de ces éléments sont encapsulées dans le bloc de contraintes des règles de production.

L'identification de ces différentes couches d'interprétation nous permet d'étendre la définition de GMC-PC en une grammaire Hiérarchisée que nous nommons **GMC-HPC (Grammaire Multi-ensembles à Contraintes Hiérarchique et Pilotée par**

le Contexte). Le couplage de la hiérarchisation des contextes et de la hiérarchisation des production constitue notre première contribution majeure sur cet axe de la thèse. Le pilotage par le contexte du formalisme de base GMC-PC implique que toutes les règles contextuellement pertinentes (au moins un CSD satisfait en précondition) sont déclenchées, à tout niveau de l'analyse, ce qui implique des calculs redondants. Il apparaît inutile d'évaluer des règles de la deuxième couche en même temps que l'analyseur évalue les règles de la dernière. Pour résoudre ce problème, et introduire de la hiérarchie dans le processus d'analyse, **nous définissons l'opérateur *RuleLayer***. Ainsi, les productions dans le formalisme étendu GMC-HPC seront donc définies comme suit :

$$\alpha \rightarrow \beta \left\{ \begin{array}{l} \text{RuleLayer} \\ \text{Préconditions} \\ \text{Contraintes} \\ \text{Postconditions} \end{array} \right\} \mid \alpha \in \mathrm{V}_N{}^+,\ \beta \in (\mathrm{V}_T \cup \mathrm{V}_N)^+$$

Cette formalisation oblige l'analyseur à ne tester que les règles qui appartiennent à la même couche d'interprétation, et cette hiérarchisation permettra de diminuer la complexité de l'analyse en diminuant le nombre de règles déclenchées. Reprenons par exemple le dessin vu dans la section 4.3, que nous répétons ici dans la figure 5.12. La notion de hiérarchie dans le formalisme étendu GMC-HPC conduit à l'arbre d'analyse présenté sur la figure 5.13. Nous pouvons remarquer que la hiérarchistation des règles



(a) Tracé dessiné

(b) Résultat de l'analyse

FIGURE 5.12 – Rappel du scénario défini dans la section 4.3

de production structure l'arbre d'analyse. Par exemple, nous voyons que tous polygones présents dans le document sont d'abord générés dans la couche 3 du processus d'analyse, avant de passer à la reconnaissance des figures géométriques particulières dans la couche

4.

Le même principe est appliqué pour le passage de couche 2 vers la couche 3. Cela permet de structurer l'arbre l'analyse, au sens où l'on passe d'une couche à une autre lorsqu'aucune autre règle appartenant à ce niveau n'est applicable. Cela implique que les règles appartenant à la couche 2 ne seront plus évaluées après le passage à la couche 3, même si elles sont toujours contextuellement valides. Nous évitons donc la complexité inhérente au pilotage par le contexte. Le processus d'analyse est désormais doublement piloté **par la hiérarchie et le contexte**. Un autre avantage de cette formalisation est

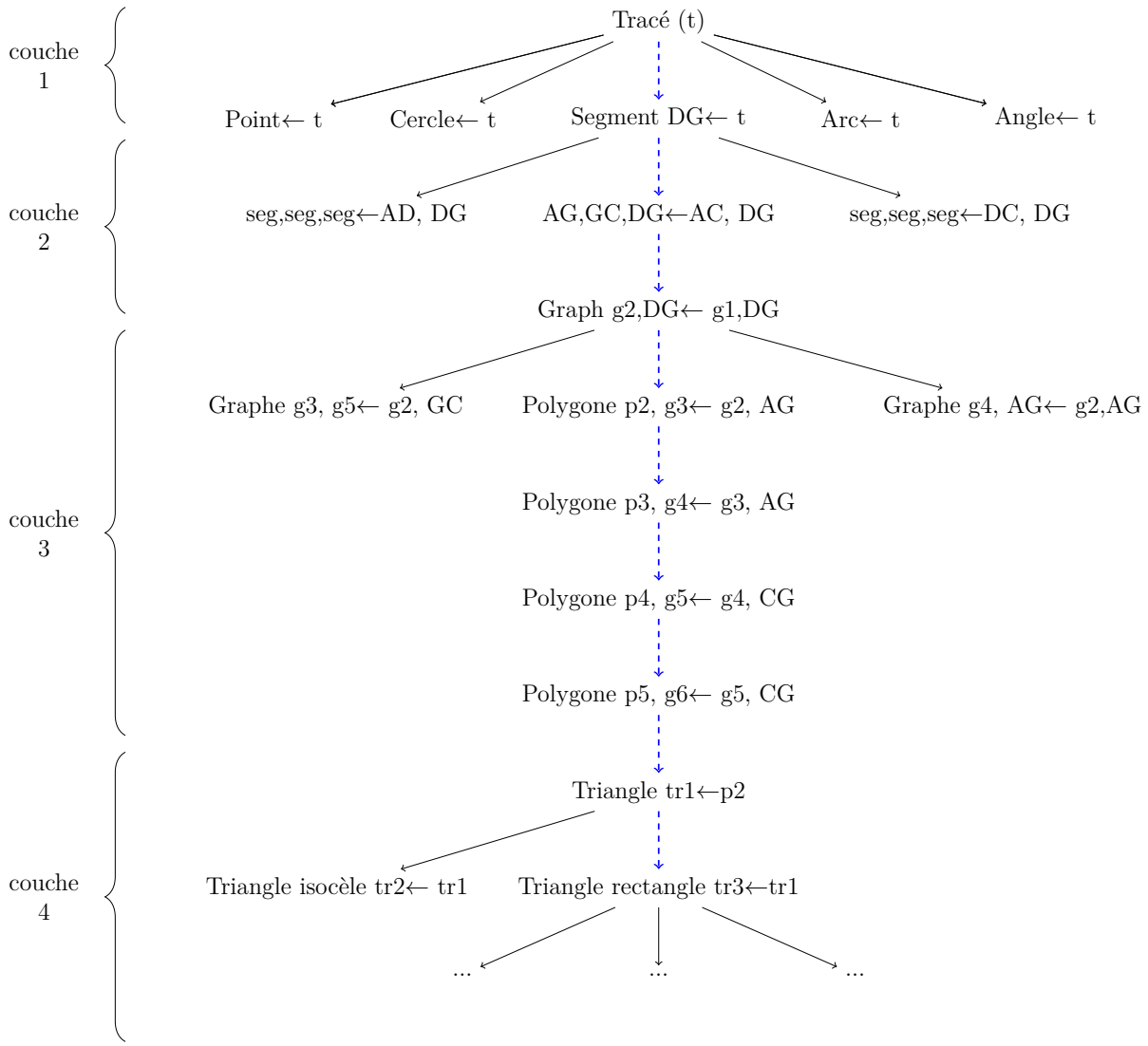


FIGURE 5.13 – Impact de la hiérarchistation des règles de production

la multiplication des stratégies d'exploration possibles par le nombre de niveaux d'interprétation du domaine d'application, puisque pour chaque couche, les deux méthodes largeur-d'abord et profondeur-d'abord sont possibles. Cette contribution ouvre l'utilisation de ce formalisme étendu à des compositions de documents complexes comme par exemple les pièces pour les plans d'architecture, les molécules pour la chimie, ou les circuits électriques pour la physique.

5.4 Analyseur Contextuel Sémantique

Nous avons évoqué dans la section 4.4 la limite inhérente au formalisme GMC-PC qui rend le processus d'analyse de figures structurés complexes : pour créer n polygones à partir de n cycles, la production *Polygone* doit être réduite n fois. Nous le voyons bien dans la figure 5.13 : la production *Polygone, graphe*$\leftarrow$ *graphe, GC* est réduite deux fois pour créer les deux polygones GCD et GCBAD. Une autre limite de ce formalisme, bien qu'il soit expressif, est qu'il est hors de portée du concepteur non développeur. La modélisation d'un graphe sémantique de connexion n'est, à notre avis, pas une solution intuitive pour l'utilisateur. Dans le cas où un expert voudrait modéliser la connaissance a priori de son domaine d'expertise, la définition des règles devrait être plus simple : par exemple, pour modéliser un polygone, la règle devrait être de la forme :

$$Polygone \leftarrow segment_1,\ segment_2, \ldots segment_n.$$

Dans cette section, nous présentons d'abord une solution à ce problème de maniabilité de la grammaire. Ensuite, nous détaillerons notre proposition d'extension du formalisme GMC-HPC par la définition d'un analyseur contextuel sémantique qui permet de remédier au problème de complexité inhérent à la formalisation des règles de production de formes à structure récursive (les graphes sémantiques de connexion), et les figures structurées complexes (les polygones et les pièces, entre autres). Ceci constitue notre deuxième contribution majeure dans cet axe de la thèse.

5.4.1 Génération automatique à partir de la définition de l'utilisateur

Le principe de base est de simplifier la tâche du concepteur, en remplaçant la définition des règles de production complexes par une simple définition des éléments de connexion, et des *CSD de connexion*.

Nous entendons par *CSD de connexion* les CSD qui permettent de déclencher les règles de production de formes à structures récursives (les graphes sémantiques) ou les figures structurés complexes (induites des patterns particuliers, *e.g.* les polygones ou les pièces). Le concepteur spécifie donc aussi le pattern désiré à détecter entre ces éléments (un cycle dans pour la géométrie et les plans d'architecture).

La figure 5.14 illustre la modélisation par le concepteur pour le domaine de la géométrie. À partir de cette définition de l'utilisateur, l'analyseur contextuel sémantique (A.C.S)

Pattern : Cycle
Type des éléments de connexion : segment
CSD de connexion : (segment : s_1) [Extrémité] (segment : s_2) [extrémité]
Type d'éléments réduits : Polygone

FIGURE 5.14 – Spécification requise pour la génération de l'analyseur contextuel sémantique

est généré automatiquement, parallèlement à l'analyseur du formalisme GMC-HPC. Plus spécifiquement, à partir de la définition des éléments et des CSD de connexion, les éléments suivants sont générées :

— Les règles de création de graphes sémantiques de connexion, explicitées dans la section 5.1.1 ;
— Le CSD de connexion relatif à la mise à jour de ces graphes :
(Segment : s)[extrémité](Graphe : g)[extrémité] ;
— La conjonction de CSD relative à la recherche de cycles :
(Segment : s)[premier](Graphe : g)[extrémité] &
(Segment : s)[dernier](Graphe : g)[extrémité]
qui modélise le fait qu'un nouveau segment est connecté par ses deux extrémités à un graphe sémantique de connexion.

Le rôle de l'analyseur contextuel sémantique réside donc dans plusieurs actions : (1) mettre à jour la structure globale du document, dès lors qu'un CSD de connexion est satisfait, et (2) rechercher des patterns particuliers (cycles) dans le graphe de connexion, (3) créer les figures structurelles complexes (polygones), et (4) les redonner en entrée à l'analyseur GMC-HPC pour continuer le processus d'analyse classique.

5.4.2 Processus d'analyse

La figure 5.15 illustre l'intégration de l'analyseur contextuel sémantique. La combinaison des éléments et des CSD de connexion déclenche la création et la mise à jour des graphes sémantiques de connexion. À chaque création/mise à jour d'un graphe, par l'ajout d'un nouveau segment qui lui est connecté, l'analyseur contextuel sémantique recherche de nouveaux cycles pour la création de polygones. Ces nouveaux cycles contiennent nécessairement le nouveau segment analysé et seront directement transformés en polygones. On voit bien ici que l'on sort de la problématique de la réduction de n règles pour la création de n polygones à partir d'un nouveau segment. La hiérarchie de ce nouveau processus d'analyse est maintenue, puisque l'analyseur contextuel sémantique ne joue un rôle que dans la couche 3, qui est la première couche sémantique. Les nouveaux éléments sémantiques seront donnés en entrée à l'analyseur de GMC-HPC. Ils seront les points d'entrées de l'analyse dans la couche sémantique suivante (couche 4) pour la création de polygones particuliers (quadrilatères, triangles...).

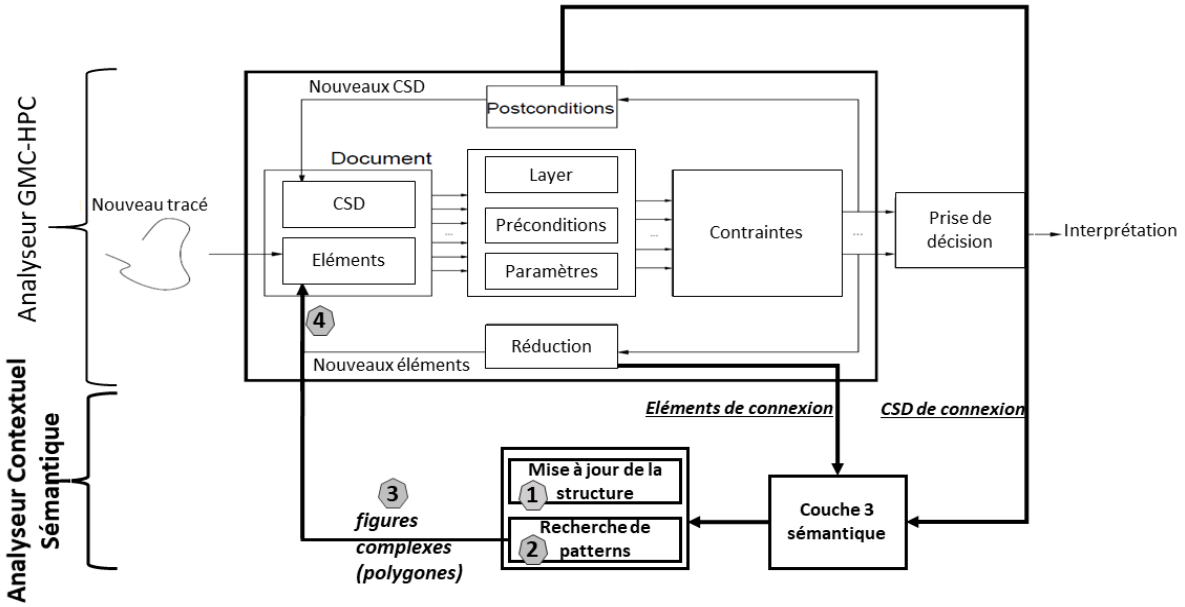


FIGURE 5.15 – Interaction entre l'analyseur GMC-HPC et l'analyseur contextuel sémantique

L'impact de l'intégration de l'analyseur contextuel sémantique est illustré dans la figure 5.16 qui présente le nouvel arbre de dérivation créé pour l'analyse du dessin de la figure 5.12 (aussi utilisé dans la section 4.3). Les polygones créés p2, p3, p4, et p5 se réfèrent aux deux triangles DGC, AGD, et aux deux polygones AGDCB, et ADGCB. La réduction

de la complexité s'explique par le fait que la règle de production de polygones ne sera plus appliquée quatre fois, comme dans la figure 5.13. En effet, le segment [GC] (il en est de même pour [AC]) va déclencher les CSD relatifs à la recherche de cycles, ce qui aura comme conséquence la génération de deux polygones DGC et ADGCB. Cette recherche se fera une unique fois, au lieu de deux fois, par l'application de la règle Polygone← Graphe, GC.

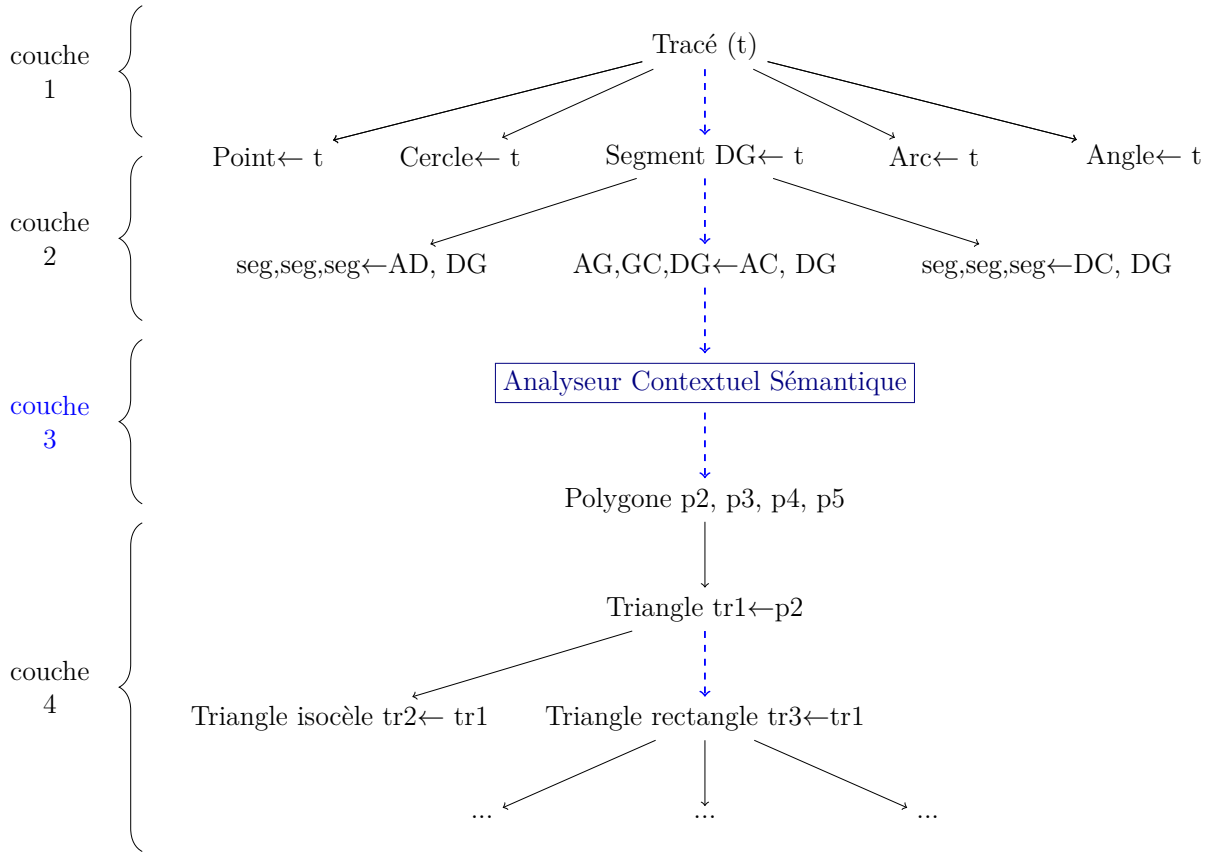
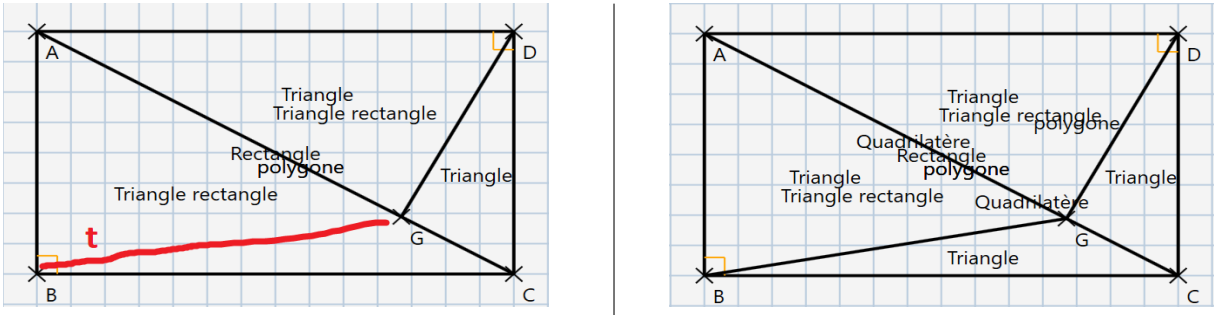


FIGURE 5.16 – Impact de l'analyseur contextuel sémantique

L'avantage de cette approche est plus frappant quand on considère un exemple un peu plus complexe, comme celui de la figure 5.17. Le nouveau segment [GB] va en effet engendrer la création de six nouveaux polygones (**GB**C, **GB**A, **GB**AD, **GB**CD, **GB**ADC, **BG**CDA). Tandis que la règle de création de polygone serait évaluée six fois dans le processus d'analyse de base du formalisme étendu GMC-HPC, elle sera effectuée uniquement une fois grâce à l'intégration de l'analyseur contextuel sémantique, réduisant par conséquent la complexité du processus.



(a) Tracé dessiné

(b) Résultat de l'analyse

FIGURE 5.17 – Ajout d'un nouveaux segment et la création de 6 polygones

5.4.3 Stratégie d'optimisation

Il convient à ce stade de distinguer deux types de polygones : ceux qui sont "explicites" (GBC, GBA), et ceux qui sont "implicites" (GBAD, GBCD, GBADC, BGCDA). Le terme "implicite" se réfère à des combinaisons de polygones adjacents (*e.g* GBAD = GBA + ADG). Cette notion "implicite" est cruciale pour maintenir les polygones créés dans le cas où un segment adjacent à deux polygones serait supprimé (par exemple, maintien de GBAD, lors de la suppression de segment [AG]). Par conséquent, même si l'intégration de l'analyseur contextuel sémantique réduit la complexité du processus d'analyse, le calcul de tous cycles devient problématique au fur et à mesure que le document se complexifie. Nous proposons une stratégie d'optimisation pour résoudre ce problème.

Recherche de cycles minimaux

Il est clair que l'augmentation de la taille de l'espace de recherche est générée par la détection des polygones implicites. Pour contrôler la complexité, la première piste évidente est de calculer seulement les cycles minimaux. Il en découle que seuls les polygones explicites seront générés. Toutefois, nous perdons la sémantique des polygones implicites, et par conséquent, la sémantique du document. Si l'utilisateur supprime le segment [AG] commun à GBA et ADG, la scène de la figure 5.17 ne contiendra que les triangles GBC et GCD. Cela veut dire que calculer seulement les cycles minimaux rend le système inutilisable. Pour régler ce problème, nous introduisons le concept de **l'analyse déclenchée par la suppression**.

Analyse déclenchée par une action de suppression

GMC-HPC, en tant que grammaire visuelle, est générative, *i.e.* l'analyse est déclenchée seulement par l'ajout d'un nouveau tracé manuscrit. Puisque la suppression d'un segment implique une mise à jour des cycles dans le graphe de connexion le contenant, un nouveau processus d'analyse devrait avoir lieu. La figure 5.18 illustre ce processus d'analyse engendré par une suppression d'un segment par l'utilisateur.

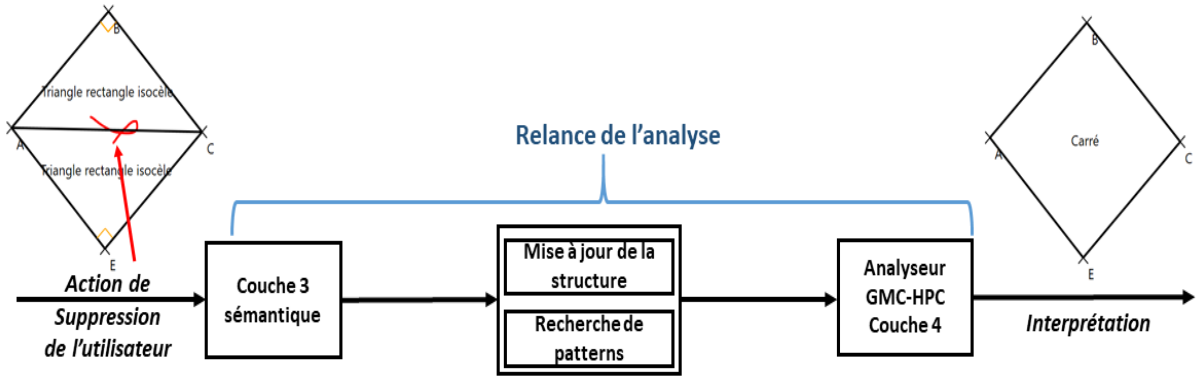


FIGURE 5.18 – Analyse déclenchée par une action de suppression

Contrairement à l'analyse classique déclenchée par un tracé manuscrit, ce processus commence au niveau de la première couche sémantique (couche 3 dans la figure 5.16). Dans cet exemple, l'utilisateur supprime le segment [AC], adjacent aux deux triangles rectangles isocèles ABC et ABE. Le graphe sémantique de connexion contenant le segment supprimé est mis à jour, et les nouveaux cycle minimaux sont détectés (A-B-C-E dans l'exemple). Les polygones créés à partir de ces nouveaux cycles sont ensuite donnés en entrée à l'analyseur GMC-HPC pour passer à la couche 4 du processus (et reconnaître un carré dans cet exemple). De cette manière, **il n'y a plus besoin de maintenir la connaissance sur les figures implicites**, puisqu'on est capable de les retrouver une fois les figures explicites supprimées par l'utilisateur (dans l'exemple, l'analyseur est capable de reconnaître un carré lors de l'action de suppression du segment [AC] et des triangles ABC et ABE). Cette dernière optimisation a un impact important en terme de complexité d'analyse : le processus d'interprétation à la volée est désormais capable de respecter la contrainte d'interaction utilisateur en temps-réel. Nous verrons l'impact de notre extension dans le chapitre expérimentation (chapitre 6).

5.5 Générécité de l'approche : application à la composition de plans architecturaux

Afin de prouver la généricité de notre approche, nous avons adapté notre extension du formalisme au domaine de l'architecture. Le formalisme de base GMC-PC a déjà été adapté à ce domaine, dans le contexte de l'interprétation à la volée [PMA10], et l'interprétation a posteriori [Gho12]. Les éléments de base ici sont les murs (l'équivalent des segments), les portes et fenêtres éditables. Nous ajoutons la capacité de reconnaître des pièces, ce qui rajoute une couche sémantique qui n'était pas présente dans les précédentes adaptations du formalisme de base à ce domaine. La figure 5.19 illustre la règle de production d'une porte, tandis que la figure 5.20 présente le processus de composition et d'interprétation à la volée dans ce domaine d'application qu'est l'architecture.

Porte : p $\rightarrow$ **tracé t** avec :
Couche : 1
Préconditions :
(Mur : m1) [LongueurSegment] (t) [premier] & (Mur : m1) [LongueurSegment] (t) [dernier]
Contraintes :
Reconnaisseur(t, Porte)
Postconditions :
(p)[Dans] (tracé : t2) [premier] $\Longrightarrow$ [Porte-Inversée $\rightarrow$ p]

FIGURE 5.19 – Règle de production d'une porte en GMC-HPC

La figure 5.21 illustre un exemple de plan complexe contenant plusieurs pièces (l'utilisateur peut renommer les pièces reconnues par l'analyseur). Chacune des actions de suppression de l'utilisateur (en rouge sur figure 5.21.a) entraînent une un processus d'analyse consistant à calculer de nouvelles pièces et mettre à jour le plan (le résultat de ces actions est illustré dans la figure 5.21.b). Notre extension s'avère nécessaire pour que la performance au regard du temps d'analyse soit acceptable pour le public cible (professionnels en bâtiment). En effet, en parallèle à notre travail sur la géométrie, cette adaptation de notre extension du formalisme GMC-PC a permis de concevoir une application de composition de plans d'architecture qui a été transférée à une société. Nous verrons par la suite, dans le chapitre 6, le bien fondé de notre contribution sur ces deux domaines d'application.

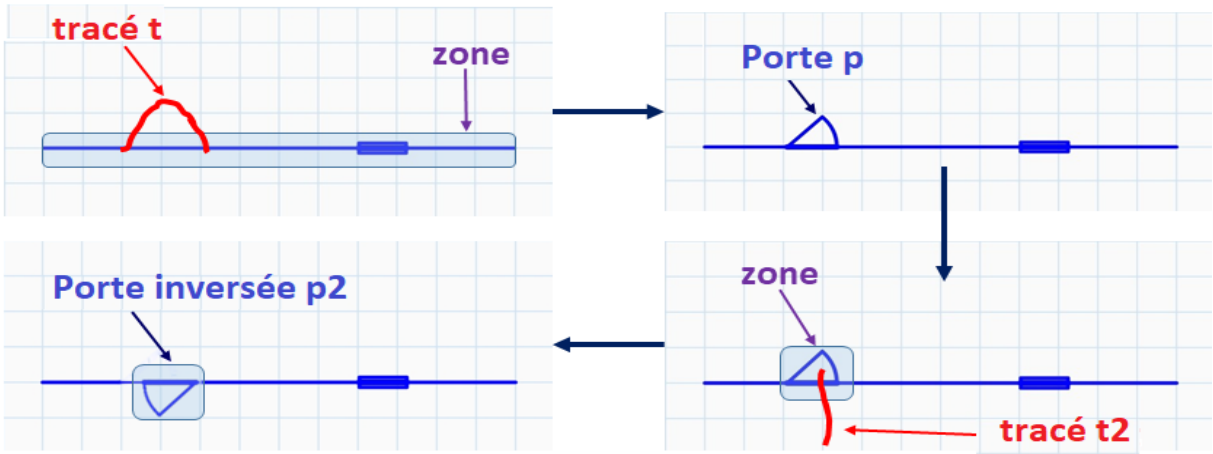
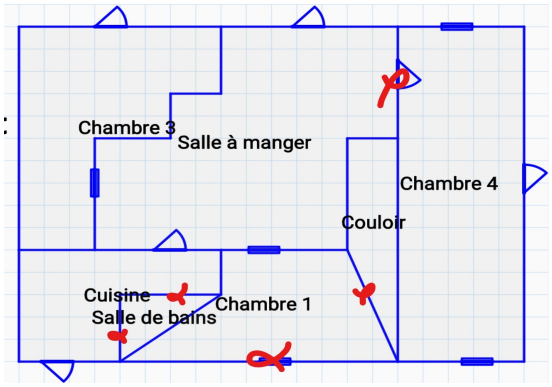
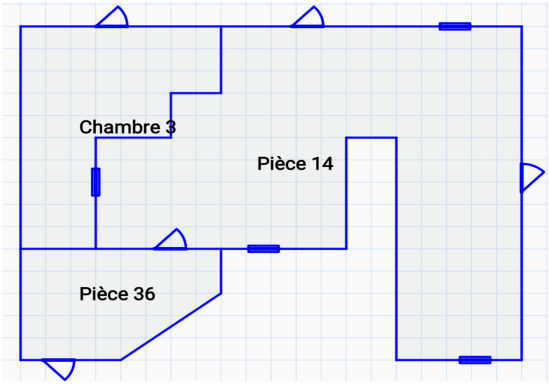


FIGURE 5.20 – Composition et interprétation à la volée de porte : création et inversion de celle-ci



(a) Actions de suppression de l'utilisateur



(b) Document après suppression

FIGURE 5.21 – Complexité du domaine de plans d'architecture

5.6 Bilan

Dans ce chapitre, nous avons présenté les différents aspects de nos contributions dans le contexte du premier axe de cette thèse : l'interprétation à la volée de tracés manuscrits. Nous avons détaillé notre extension du formalisme de base GMC-PC en grammaire hiérarchique GMC-HPC. Pour ce faire, nous avons introduit la notion de hiérarchie sous deux aspects :

— Hiérarchisation des contextes : définition d'un nouvel opérateur *FirstContext* qui permet de choisir le premier contexte valide, et donc proposer une stratégie de recherche alternative au sein des préconditions pour accélérer l'analyse ;

— Hiérarchisation des règles de production : redéfinition des règles en ajoutant un opérateur *RuleLayer* qui permet de piloter l'analyse en combinant contexte et couche d'interprétation.

Nous avons ensuite présenté notre externalisation de la recherche du contexte global sémantique du document par la définition d'un analyseur contextuel sémantique qui est responsable du maintien des liens de connexions entre les éléments d'intérêt spécifiés par le concepteur. L'intégration de cet analyseur sémantique au processus d'analyse permet de diminuer la combinatoire relative à la recherche de patterns particuliers dans le document, en l'occurrence des cycles pour la création de polygones le domaine de la géométrie mais aussi celui de l'architecture. Dans le chapitre 6 qui suit, nous démontrerons la pertinence et la générécité de notre extension sur nos domaines d'application.

Chapitre 6

Expérimentations et résultats relatifs à l'extension du formalisme

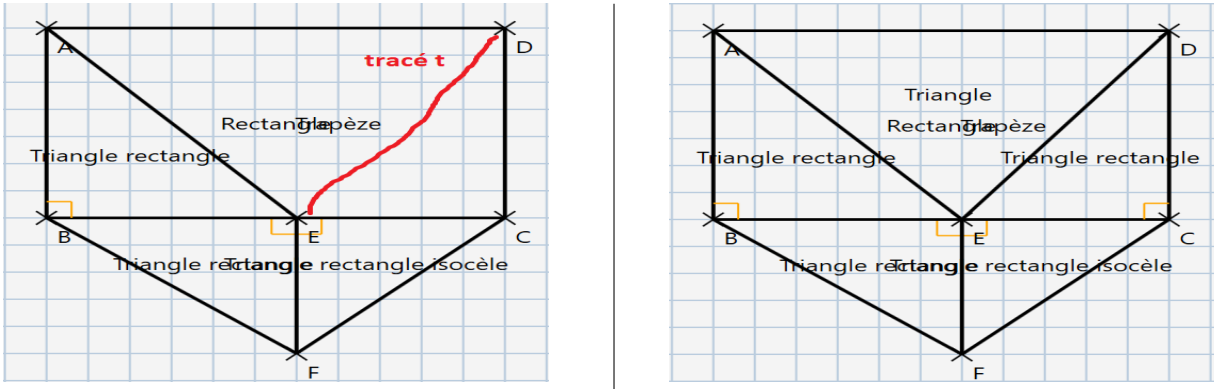
Nous étudions dans ce chapitre la pertinence de notre extension du formalisme grammatical et l'impact sur le temps d'analyse. Ce temps-d'analyse est une élément crucial pour la mise en oeuvre d'une interaction temps-réel avec l'utilisateur. L'objectif sous-jacent dans la diminution du temps-d'analyse, est d'être en capacité à générer du feedback immédiat. La production de feedback en temps réel est intéressante dans un contexte pédagogique tel que celui de la composition de figure géométrique, mais aussi dans tout contexte d'utilisation d'application interactive, car ces feedbacks permettent d'informer l'utilisateur de la bonne interprétation de ses gestes par le système. Nous évaluons notre approche sur son temps d'analyse car il représente l'attente de l'utilisateur entre le moment où il dessine un nouveau tracé et le moment où il reçoit le feedback visuel retourné par le système. Dans ces expérimentations, nous avons donc testé notre grammaire étendue GMC-HPC, pour le domaine de la composition de figure géométrique (cible du projet ACTIF), mais aussi pour le domaine de la composition de plans architecturaux. Nous ne présenterons pas de résultats sur la reconnaissance de formes, étant donné que les performances en la matière du formalisme de base GMC-PC ont été démontrées auparavant dans [MA09], et que notre contribution consiste en l'extension de la grammaire pour contrôler la complexité du processus d'analyse. Le protocole d'évaluation suivi s'inspire de ce qui a été fait dans [PMA10], et consiste à évaluer l'impact de nos contributions en comparant leurs performances à celles du formalisme de base sur des scénarios de composition de figures complexes pour lesquelles les limites de la grammaire GMC-PC sont atteintes. Les expérimentations ont été réalisées sur un PC windows 64bits, Intel i7, 2.90 Ghz et 16GB RAM. Les facteurs clés pour l'évaluation de la complexité du processus d'interprétation et la performance en terme de temps d'analyse sont les suivants :

— Itérations : nombre de règles réduites ;
— Interprétations : nombre de branche dans l'arbre d'analyse
— Temps : le temps d'analyse ;
— Déclenchées : le nombre de règles déclenchées par l'analyseur, qu'elles soient réduites ou non ;

Nous nous baserons donc sur ces critères pour évaluer l'impact de nos contributions.

6.1 Impact de la hiérarchisation des contextes

La hiérarchisation des contextes, représentée par la définition de l'opérateur First-Context (présenté dans la section 5.2), permet d'éviter de multiplier les interprétations équivalentes, qui rajoutent une complexité artificielle au processus d'analyse. Nous étudions l'impact de cet opérateur sur le scénario de composition d'un schéma de géométrie illustré dans la figure 6.1. Dans ce scenario, le tracé t sera transformé en un segment [DE], ce qui aura comme conséquence la création de 7 figures : le triangle rectangle **ED**C, le triangle **ED**A, le trapèze **ED**AB, le trapèze **ED**CF, et les polygones **ED**CFB, **ED**ABF, et **ED**CFBA. Le tracé t est lié par ses extrémités à [AD], [DC], [EF], [EC], [EA], et [EB]. Par conséquent, nous avons huit contextes cohérents pour la création du segment [ED]. Il en découle huit branches créées dans l'arbre d'analyse du formalisme de base (situation similaire à l'exemple présenté dans la section 5.2, figure 5.8).



Tracé dessiné | Résultat de l'analyse

FIGURE 6.1 – Premier scénario de construction

Le tableau 6.1 illustre les résultats de l'analyse pour ce premier scénario.

Le résultat principal que nous pouvons voir dans ce tableau est que le formalisme de

TABLE 6.1 – Impact de la hiérarchisation des contextes

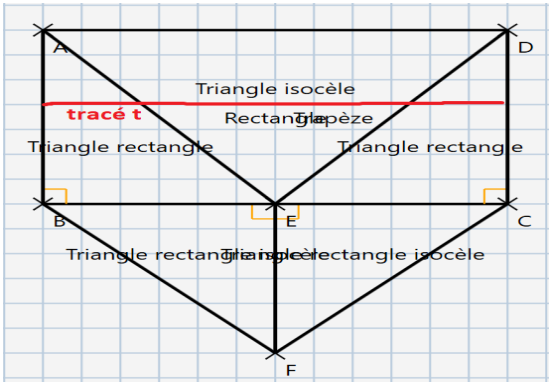
Approche	Itérations	Interprétations	Temps	Déclenchées
GMC-PC	88	8	3s 25"	1126
GMC-PC + *FirstContext*	11	1	**0s 27"**	169

base GMC-PC ne permet pas de respecter la contrainte d'interprétation à la volée. En effet, comme indiqué dans [NIE93], le processus de composition de l'utilisateur n'est pas "perturbé" si le système réagit en moins d'une seconde. Aussi, avec un temps d'analyse qui dépasse de peu les trois secondes, l'analyse basée sur GMC-PC n'est pas transparente à l'utilisateur et ne satisfait donc pas à notre objectif. Nous pouvons voir que ce temps d'analyse élevé est causé par un grand nombre de règles déclenchées (1126). Finalement, grâce à la priorisation des contextes représentée par l'opérateur FirstContext, nous réussissons à réduire drastiquement le temps d'analyse (0.27s). Nous pouvons voir que cette optimisation permet justement d'élaguer l'arbre d'analyse en éléminant les hypothèses équivalentes, ce qui diminue grandement le nombre de règles déclenchées par l'analyseur (957 règles en moins).

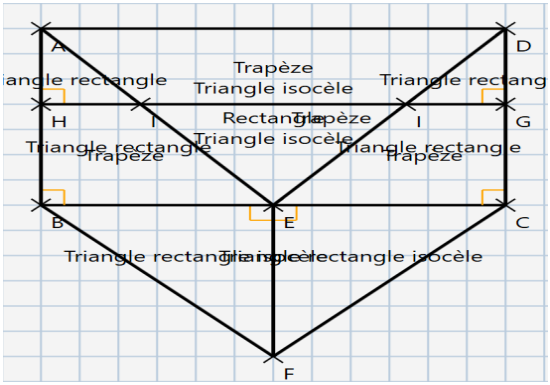
6.2 Impact de la hiérarchisation des règles de production

Outre la hiérarchisation des contextes, nous avons vu que le deuxième aspect de la définition de la hiérarchie dans le formalisme étendu GMC-HPC consiste à établir plusieurs niveaux/couches logiques d'interprétation, en utilisant le nouvel opérateur *RuleLayer*. Rappelons qu'avec cette hiérarchisation, une règle est définie par la couche logique à laquelle elle appartient. Seules les règles appartenant à la même couche peuvent être déclenchées à une phase d'analyse donnée. Cela revient à restructurer l'arbre d'analyse, et à guider le processus d'analyse par la hiérarchie des règles de production, pour contrôler la combinatoire. Nous étudions l'impact de la hiérarchisation des règles dans le scénario de construction complexe de géométrie illustré dans la figure 6.2.

Le tracé t sera interprété comme segment [HG]. L'intersection de [HG] avec [AE] et [DE] aura comme conséquence les réductions de règles de division et la création des segments [HI], [IJ], et [JG]. La mise à jour des connexions dans le document induira la création de 62 polygones (*e.g.* BHIE, BADJIE, BAIJDCFE...). Ce scénario permet d'illustrer le



Tracé dessiné



Résultat de l'analyse

FIGURE 6.2 – Deuxième scénario de construction

déclenchement de plusieurs règles appartenant à différents niveaux d'interprétation. Le tableau 6.2 présente nos critères d'évaluation de la hiérarchisation en couches logiques sur le processus d'analyse. Les deux premières lignes présentent les résultats obtenus en utilisant le formalisme de base (GMC-PC), et la première optimisation (GMC-PC + FirstContext). La troisième ligne présente les résultats du formalisme étendu *GMC-HPC*, défini par la combinaison de la hiérarchisation des contextes (*FirstContext*) et la hiérarchisation des règles de production.

TABLE 6.2 – Impact de la hiérarchisation des règles de production

Approche	Itérations	Interprétations	Temps	Déclenchées
GMC-PC	81	1	3s 79”	1246
GMC-PC + $FirstContext$	81	1	3s 75”	1246
GMC-HPC	81	1	**1s 92”**	1081

Encore une fois, la grammaire GMC-PC est dans l'impossibilité d'analyser le tracé en temps-réel, avec une durée d'analyse de 3.79 secondes. La présence d'une seule branche d'analyse dans l'arbre à la base fait que l'opérateur $FirstContext$ n'apporte rien en terme de réduction de la combinatoire dans ce cas (même nombre de règle déclenchées et même temps d'analyse = 3s 75”). Cela s'explique par le fait que $FirstContext$ a un impact sur la performance lorsqu'il y a plusieurs contextes cohérents possibles, ce qui n'est pas le cas ici (le tracé t n'est connecté à aucune zone d'extrémité). C'est la hiérarchisation des productions dans le formalisme étendu qui permet de réduire l'espace de recherche des règles et ainsi de diminuer de 165 le nombre de règles déclenchées. Cette diminution

permet d'avoir un temps d'analyse de 1s 91".

L'extension de la définition de GMC-PC en grammaire hiérarchique GMC-HPC permet donc de limiter la complexité du processus d'analyse. Le scénario que nous avons étudié dans la figure 6.2 est particulièrement complexe et n'est pas très fréquent. Dans le contexte de la composition de figures géométriques, la performance est acceptable, surtout si la durée d'analyse ne dépasse la seconde que dans de rares cas. Cependant, en dehors du contexte de l'apprentissage de la géométrie, pour les plans architecturaux par exemple, ce type de figures complexes devient assez récurrent. Comme évoqué précédemment, pour ne pas perturber un utilisateur, il faudrait que le temps de réaction du système soit inférieur à une seconde. Hors, il est clair qu'au fur et à mesure que le document se complexifie, la combinatoire augmente jusqu'à exploser. Cette explosion combinatoire est liée à la définition des règles de production de formes structurées complexes (*e.g.* polygones pour la géométrie, pièces pour l'architecture) : pour créer n polygones/ pièces à partir de n cycles, la production *PolygonePièce* doit être réduite n fois. Donc dans l'exemple présenté dans la figure 6.2 et le tableau 6.2, la grande majorité des itérations (81) est dédiée à la création de 62 polygones. Pour répondre à ce problème, nous avons mis en place l'*analyseur contextuel sémantique* (*cf.* section 5.4) qui va permettre d'exernaliser la recherche de patterns particuliers (les cycles dans le cas de la géométrie et de l'architecture).

6.3 Impact de l'analyseur contextuel sémantique

La composition de plans architecturaux est un domaine où l'élaboration de figures complexes est bien plus fréquente que lors de leçons de géométrie. Nous étudions donc l'impact de l'intégration de l'analyseur contextuel sémantique sur deux scénarios de composition de plans (*cf.* figure 6.3, figure 6.4), avec une complexité d'analyse progressive.

Le tableau 6.3 présente l'impact de cette contribution pour le scénario de la figure 6.3, en la comparant à la performance du formalisme étendu GMC-HPC avec :

— ACS se réfère à l'analyseur contextuel sémantique "classique" : la connaissance sur les figures "implicites" est maintenue (besoin de calculer tous les cycles engendrés par un segment ;

— OPT se réfère à l'optimisation de l'analyse déclenchée par la suppression : calcul des cycles minimaux, génération des figures "explicites" uniquement lors la création des polygones/pièces tout en introduisant le concept de l'analyse déclenchée par la suppression, afin de ne pas perdre la sémantique des figures "implicites" (*c.f*

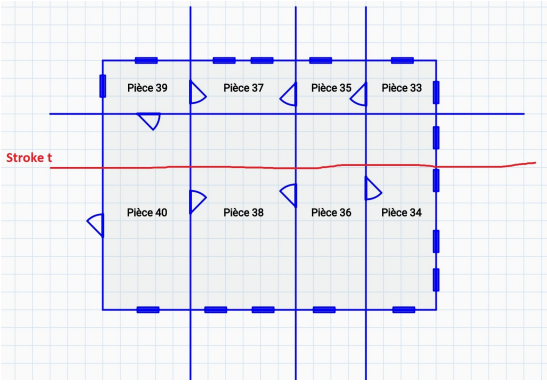


FIGURE 6.3 – Troisième scénario

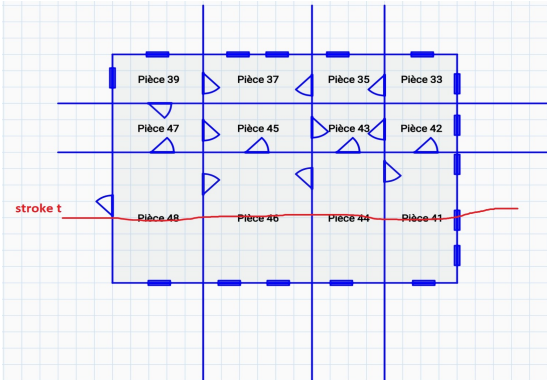


FIGURE 6.4 – Quatrième scénario

sous-section 5.4.3).

TABLE 6.3 – Résultats de l'analyse du troisième scénario (basé sur la figure 6.3)

Approche	Itérations	Interprétations	Temps	Déclenchées
GMC-HPC	788	1	1min 54s	5034
GMC-HPC +ACS	12	1	8s	1356
GMC-HPC +ACS + OPT	12	1	0s 35"	407

Notons que les 788 itérations lors de l'évaluation de GMC-HPC se réfèrent à la création de 779 pièces implicites et explicites, en plus des règles réduites appartenant aux autres couches d'interprétation. L'intégration de l'analyseur contextuel sémantique améliore de façon très significative la performance en termes d'itérations mais aussi de temps d'analyse (de 1 minute et 54 secondes à 8 secondes), grâce à la diminution des règles déclenchées (3678 en moins). Toutefois, le temps d'analyse est toujours supérieur à la seconde. C'est finalement l'optimisation OPT qui permet au système d'atteindre un temps inférieur à la seconde, et ainsi d'avoir une performance acceptable de 0s 35".

Le tableau 6.4 présente le résultat des systèmes pour le scénario illustré dans la figure 6.4, qui correspond un cas d'explosion combinatoire due à la complexification du document, si l'on ne met pas en place les extensions du formalisme présentées.

Nous pouvons voir dans ce cas qu'il n'est pas envisageable de proposer un système orienté stylet pour la composition de plans d'architecture basé sur GMC-HPC, le temps d'analyse du tracé étant de l'ordre de 5 heures, avec 350151 règles déclenchées pour la création de 6429 polygones. Le besoin d'externaliser l'analyse des patterns particuliers (cycles/ polygones/ pièces) devient évident. Bien que l'intégration de l'analyseur contextuel sémantique réduit significativement la durée à trois minutes. C'est son couplage avec

Table 6.4 – Résultats de l'analyse du quatrième scénario (basé sur la figure 6.4)

Approche	Itérations	Interprétations	Temps	Déclenchées
GMC-HPC	6442	1	5h 0min 3s	305151
GMC-HPC +ACS	12	1	2min 58s	6987
GMC-HPC +ACS + OPT	12	1	0s 37"	544

l'optimisation finale *OPT* qui permet d'avoir une performance acceptable (moins de 1 seconde : 0s 37") dans le cadre d'une interaction en temps réel avec l'utilisateur.

6.4 Discussion

Nous avons démontré dans ce chapitre l'impact très significatif de l'extension de la définition de la grammaire bi-dimensionnelle GMC-PC en grammaire hiérarchique, ainsi que celui de l'intégration d'un analyseur contextuel sémantique au processus d'analyse afin de contrôler sa combinatoire. Ces résultats ont été publiés dans [2] [3] [5] et [6]. Plus généralement, nous avons répondu dans ce premier axe de la thèse à notre premier objectif qui était l'interprétation à la volée des tracés de l'utilisateur, et plus spécifiquement l'interprétation de figures structurées complexes manuscrites. L'intérêt de nos contributions est qu'elles préservent l'aspect générique de la grammaire. Elle permettent aussi d'ajouter à l'analyse structurelle du document une analyse sémantique par la définition de la notion de la multiplicité des couches logiques d'interprétation. L'externalisation de la recherche de patterns particuliers permet de résoudre les explosions combinatoires de l'analyseur et de faciliter la manipulation de GMC-HPC par les concepteurs non développeurs. Il faudrait bien sûr penser à diversifier les patterns que le concepteur peut spécifier dans la grammaire. Les cycles sont suffisants pour la géométrie, les plans d'architecture, et la composition de circuits électriques par exemple, mais le concept pourrait être généralisé à d'autres documents structurés, comme par exemple les diagrammes de chimie moléculaire. Le formalisme étendu et l'analyseur associé sont la base de ce qu'on a appelé le *moteur de reconnaissance 2D*. Ce moteur a été validé dans les publications suivantes : [2] [3] [5] [6].

Dans la prochaine partie de ce manuscrit, nous nous intéresserons au deuxième axe de cette thèse : la supervision interactive d'exercices de construction en géométrie.

Troisième partie

Tutorat et supervision d'exercices de construction en géométrie

Chapitre 7

Principes généraux du tuteur

Nous avons présenté dans la partie précédente le premier axe de cette thèse, consistant à définir le formalisme grammatical étendu GMC-HPC, qui sera responsable de l'interprétation des tracés. Cette connaissance est encapsulée dans *le moteur de reconnaissance 2D*. Nous nous intéressons maintenant au deuxième axe de notre travail qui concerne le tutorat et la supervision d'exercices de construction en géométrie. La figure 7.1 illustre l'architecture d'IntuiGeo (que nous avions évoquée dans le chapitre 3). Comme dit précé-

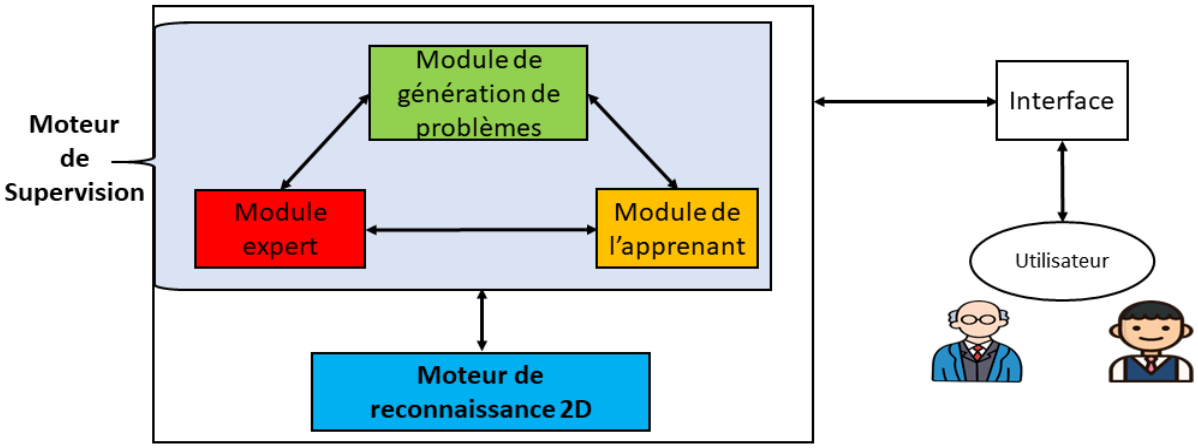


Figure 7.1 – Architecture d'IntuiGeo

demment, l'aspect tutoriel d'IntuiGeo est encapsulé dans **un moteur de supervision**. Ce moteur se base sur trois principes différents, représentés par les trois modules qui le composent :

— La génération automatique de problèmes de constructions à partir d'un exemple dessiné par l'enseignant (*module de génération de problèmes*) ;

— La supervision et l'évaluation de la résolution de l'élève pour générer des feedbacks de correction (*module de l'apprenant*) ;

— La capacité de synthétiser des stratégies de résolution pour générer des feedbacks de guidages (*module de l'expert*).

Nous allons donc détailler dans cette partie du manuscrit les différents rôles et spécificités de ces modules, en mettant en évidence l'interaction entre le **moteur de reconnaissance 2D** et le **moteur de supervision**. Plus spécifiquement, dans ce chapitre, nous nous intéressons aux principes de base de notre moteur de supervision en nous focalisant sur les différents scénarios d'interaction entre l'utilisateur (enseignant/élève) et le tuteur.

7.1 Génération d'exercices à partir d'un exemple solution

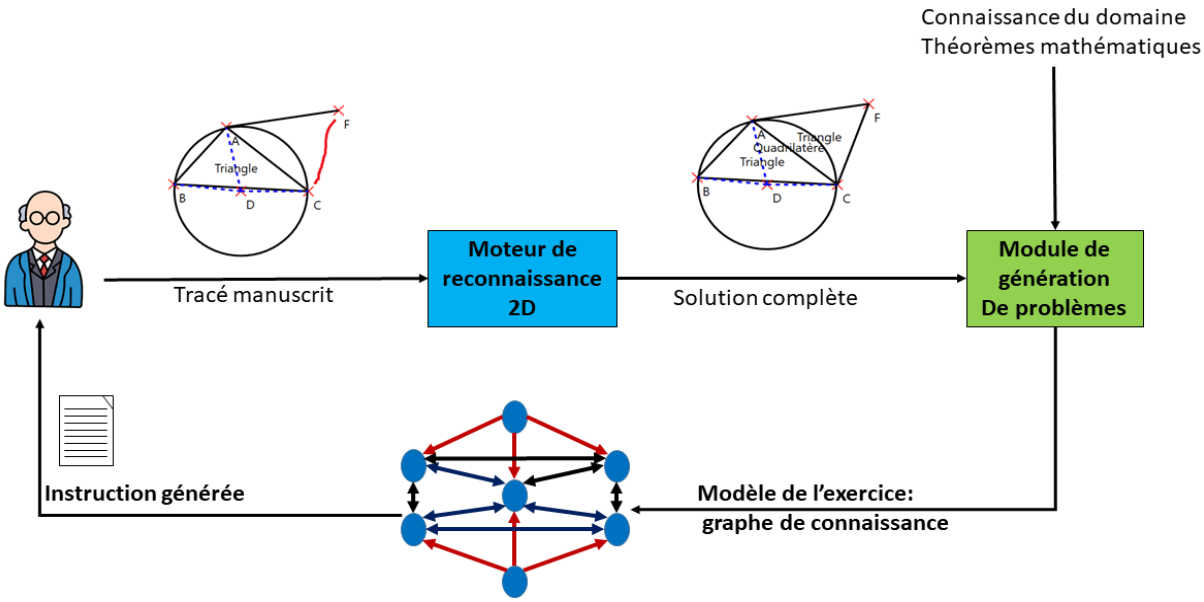


FIGURE 7.2 – Interaction de l'enseignant avec le tuteur intelligent

Le processus de création d'un nouveau problème de construction géométrique par l'enseignant est détaillé dans la figure 7.2. L'enseignant dessine un exemple complet de la solution du problème sur l'interface d'IntuiGeo. Les tracés manuscrits sont interprétés à la volée par le moteur de reconnaissance 2D. Quand la figure cible est réalisée, l'enseignant le signifie au tuteur intelligent (en appuyant sur un bouton "créer exercice"). Cela déclenche **le module de génération de problèmes** qui prend en entrée d'une part les éléments géométriques interprétés par le moteur de reconnaissance 2D et, d'autre part, la connaissance du domaine (*i.e.* les théorèmes de mathématiques relatifs à la géométrie euclidienne). À partir de ces éléments, il crée un modèle de l'exercice, sous forme

de **graphe de connaissance** [Yan+18]. Ce graphe contiendra les entités géométriques de la production de l'enseignant et les relations/contraintes mathématiques qui les lient. Les contraintes du problème sont déduites des théorèmes mathématiques intrinsèques à la définition des éléments présents. Par exemple, si l'enseignant dessine un triangle rectangle ABC en A, le module applique le théorème : $ABC \text{ rectangle en } A \implies [AB] \perp [AC]$, et met à jour le graphe de connaissance en créant le lien Perpendiculaire ([AB], [AC]). **Cette modélisation permet de s'affranchir de la procédure suivie par l'enseignant, puisque toutes les solutions possibles au même problème doivent satisfaire, *in fine*, les mêmes contraintes mathématiques**. Le module de génération est aussi capable d'extraire une instruction (consigne) du problème à partir du graphe de connaissance, instruction modifiable par l'enseignant, par ailleurs. La construction du graphe de connaissance sera présentée en détail dans le chapitre 8.

7.2 Supervision et évaluation : modélisation de l'élève

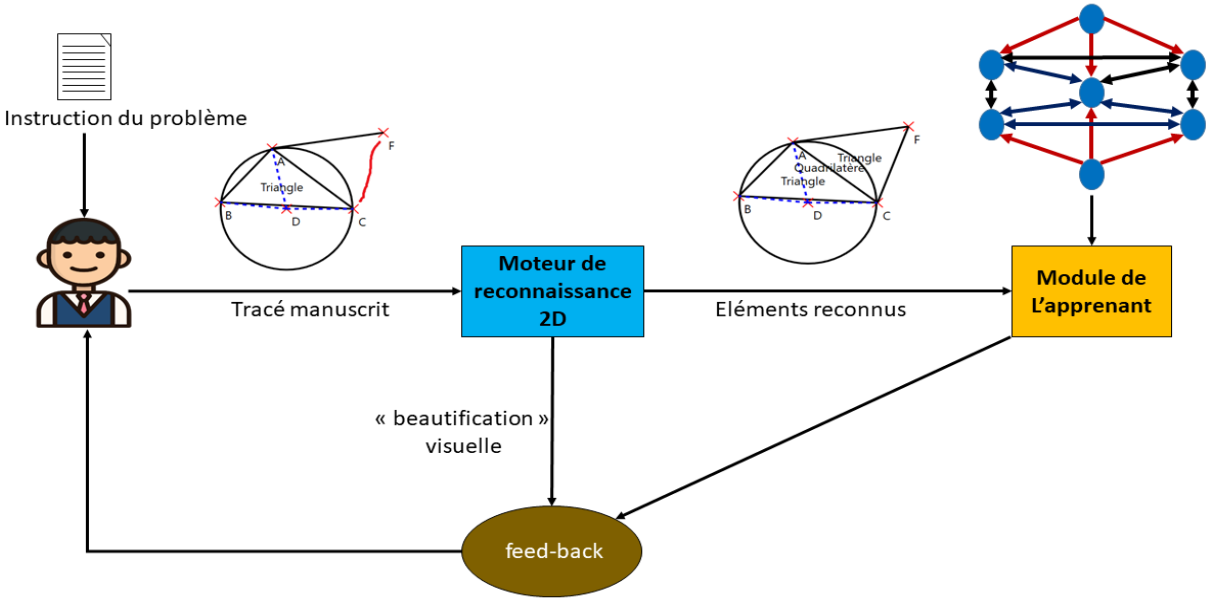


FIGURE 7.3 – Interprétation et évaluation à la volée des actions de résolution de l'élève par le **module de l'apprenant** en vue de générer des **feedback de correction**

L'interaction de l'élève avec le tuteur intelligent dans le contexte de la résolution d'un problème de construction est détaillée dans la figure 7.3. Comme pour l'enseignant, chaque tracé de l'élève est interprété à la volée par le moteur de reconnaissance 2D. À

la différence du mode auteur présenté pour l'enseignant où la figure doit être complète pour être donnée au module de génération, là chaque tracé est donné en entrée au module de l'apprenant, qui essaiera d'interpréter et d'évaluer l'action de l'élève en temps-réel. L'interprétation de cette action consiste à mettre en correspondance l'élément géométrique reconnu avec un des noeuds du graphe de connaissance, construit auparavant par le module de génération de problèmes, et de mettre à jour l'état de ce noeud. Une fois le noeud approprié trouvé, il est possible de vérifier si toutes les contraintes qui lui sont associées sont vérifiées. En s'appuyant sur l'extraction des contraintes satisfaites et non satisfaites, le tuteur est capable de générer un feedback de correction qui peut prendre plusieurs formes (feedback de couleur ou feedback textuel). Ce feedback de correction est dit descriptif puisqu'il consiste à donner des informations sur les erreurs (les contraintes non satisfaites), sans pour autant indiquer à l'élève comment les résoudre. Le module de l'apprenant sera présenté en détail dans le chapitre 8.

7.3 Synthèse de stratégies de guidage : modélisation de l'expert

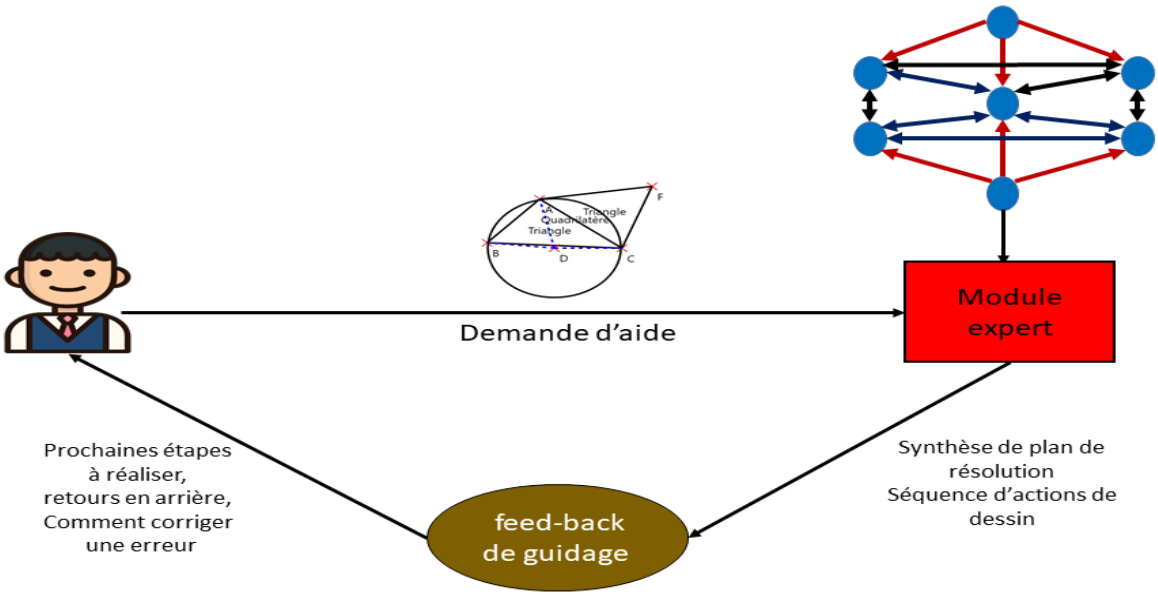


FIGURE 7.4 – Interaction de l'élève avec le module expert : à partir d'une demande d'aide explicite, le tuteur synthétise des stratégies de résolution pour fournir des feedbacks de guidage

La figure 7.4 présente l'interaction de l'élève avec le tuteur dans le contexte de la génération de **feedback de guidage**. En effet, lorsque l'élève est dans une impasse, il a la possibilité de demander explicitement de l'aide au tuteur. Cette aide est générée grâce au **module expert**. La différence entre feedback de correction et de guidage réside dans le fait que le guidage donne une piste sur la procédure à suivre (soit pour corriger une erreur, ou avancer dans la résolution), alors que le feedback de correction consiste à mettre en évidence les erreurs commises, sans plus de détail. Pour générer le feedback de guidage, le module expert se base sur le graphe de connaissance, en particulier sur l'état des contraintes, pour trouver une solution au problème. En effet, l'état du graphe représente l'état de résolution de l'élève, le module expert essaie donc de synthétiser des stratégies pour arriver à un état où toutes les contraintes sont satisfaites. Pour cela, comme nous le détaillerons dans le chapitre 9, nous reformulons le problème de résolution en *un problème de planification* tel que l'état initial correspond à une zone de dessin vide et l'état final/but correspond à une production où toutes les contraintes dans les relations du graphe de connaissance sont satisfaites. Le but est donc de trouver un plan solution, *i.e.* une séquence d'actions qui mènent d'un état du graphe où une ou plusieurs contraintes sont non satisfaites, à l'état final souhaité. La connaissance de l'expert est définie par un ensemble d'actions de dessin qui modélisent la connaissance procédurale du domaine de la géométrie de construction. Le plan solution généré par le module expert est traduit à l'élève sous forme textuelle et peut comprendre des indications sur les prochaines étapes à réaliser, des conseils sur des retours en arrière nécessaires à la résolution, ou encore des indications détaillées permettant de savoir comment résoudre une erreur (comment utiliser les outils par exemple).

7.4 Bilan

Par cette modélisation par contraintes de la connaissance *déclarative* (*i.e.* la connaissance relative au savoir sur une notion) et de l'état de résolution, IntuiGeo pourrait être classé comme tuteur à base de contraintes (*cf.* section 3.2.1). Cependant, une distinction importante par rapport à cette famille de tuteurs réside dans la présence d'un module expert, capable de modéliser la connaissance *procédurale* (*i.e.* relative au savoir-faire), et de générer des feedbacks de guidage à l'élève, caractéristique exclusive aux tuteurs à base de règles (*cf.* section 3.2.2). Nous pensons que cette hybridation est la plus adaptée à notre domaine. En effet, la géométrie de construction, au contraire de la démonstration

de preuves par exemple, permet trop de liberté d'actions et de stratégies à l'élève pour que le tuteur soit capable de modéliser toutes les erreurs de l'élève par des règles faussées (*cf.* section 3.2.1). Nous pensons cependant qu'il est nécessaire d'avoir la capacité de guider les élèves en cas de blocage et c'est cela que nous avons défini et intégré au moteur de supervision, le module expert, dont la connaissance est modélisée par des règles de planification.

Les chapitres suivants décrivent précisément les différents modules du moteur de supervision ainsi que les éléments dont ils ont besoin.

Chapitre 8

Modélisation de la connaissance déclarative spécifique au problème et modélisation de l'élève

Nous avons présenté dans le chapitre 7 la structuration et les principes de notre système tutoriel intelligent, avec la conception d'un moteur de supervision qui communique avec un moteur de reconnaissance 2D pour la création de problèmes et la supervision de stratégies de résolution. Dans ce chapitre, nous nous intéressons plus précisément au modèle de l'exercice, *i.e.* le graphe de connaissance, qui représentera la connaissance spécifique d'un problème. Nous détaillons le processus de génération de ce graphe par le module de génération de problèmes à partir d'un exemple de solution dessiné sur l'interface par un enseignant. Ensuite, dans la section 8.3, nous décrivons le rôle de ce graphe dans la représentation de l'état de résolution de l'élève en nous intéressant au module de l'apprenant.

8.1 Génération du modèle à partir d'exemple

Comme explicité dans le chapitre précédent, le graphe de connaissance est créé par le module de génération de problèmes à partir d'un exemple complet dessiné par l'enseignant. Il est défini comme suit :

Definition 8.1.1. Un graphe de connaissance est un ensemble de noeuds et d'arcs tels que :

- un noeud n est défini par :
  - E : un élément reconnu par le moteur de reconnaissance ;

  - *Ref* : l'ensemble des contraintes réflexives sur E, *e.g.* la longueur de E ;
  - *Def* : l'ensemble des contraintes dépendantes de la définition mathématique de E ;
  - *Struct* : l'ensemble des contraintes structurelles telles que l'adjacence, l'intersection ;
  - *État* = {défaut, partiel, complet}.
- Chaque arc est un triplet *a = (Np,Nd,Rel)* avec :
  - *Np* : le premier noeud ;
  - *Nd* : le deuxième noeud ;
  - *Rel* : l'ensemble des contraintes liants Np à Nd.

L'élément *E* (le tracé transformé en objet géométrique) est présent sous deux aspects :

1. L'objet physique (*[AB], Cercle(C), ABD...*), d'où sont déduites les contraintes réflexives *Ref* ;
2. Le concept géométrique dont découlent les contraintes de définition *Def* : par exemple pour un parallélogramme ABCD, le concept implique (entre autres) les contraintes *Def* suivantes : *Parallèle(AB,CD), Parallèle(AD,BC), Égal(AB,CD), Egal(AD,BC).*

Par la suite, nous considérerons l'union des ensembles de contraintes ($Ref \cup Def$) comme étant l'ensemble des contraintes sémantiques, que nous noterons *Sem.* L'élément E se réfère au tracé dessiné par l'enseignant dans l'exemple solution. Il est dans l'état *défaut* lorsqu'il n'a pas encore été mis en correspondance avec un tracé de l'élève. Il est dans l'état *partiel* lorsque l'élément E a été mis en correspondance avec un tracé de l'élève mais que l'une de ses contraintes est non satisfaite. Enfin, il est dans l'état *complet* lorsqu'il a été mis en correspondance avec un tracé de l'élève et que toutes ses contraintes sont satisfaites. Il apparaît donc que pour que l'exercice soit résolu, il faut que tous les noeuds du graphe soient à l'état *complet.*

8.2 Construction du graphe de connaissance

En s'appuyant sur la définition formelle d'un graphe de connaissance, voyons maintenant comment celui-ci est construit pour un exemple de figure particulière. Comme indiqué précédemment cette construction est liée à l'interaction de l'enseignant avec notre moteur de supervision et plus spécifiquement avec le module de génération de problèmes. La figure 8.1 illustre le processus de construction du graphe. L'enseignant dessine deux triangles

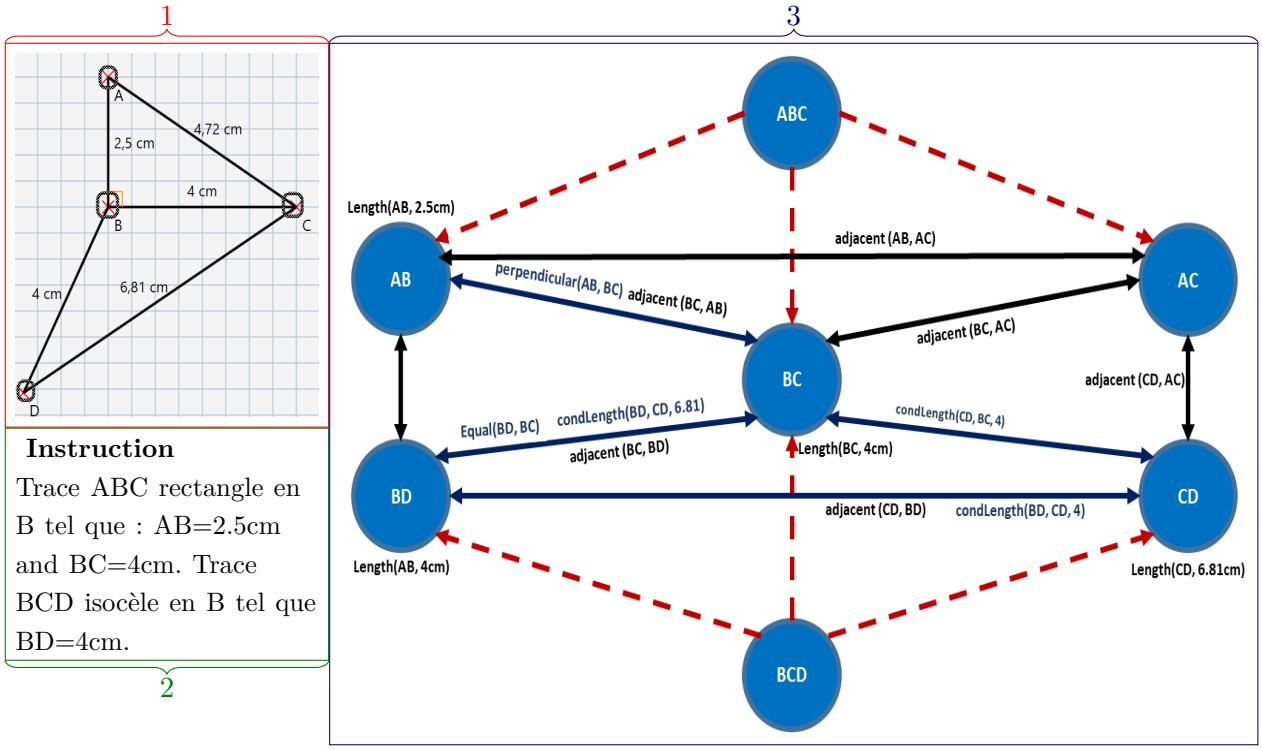


FIGURE 8.1 – Solution de l'enseignant, Construction du graphe, Instruction

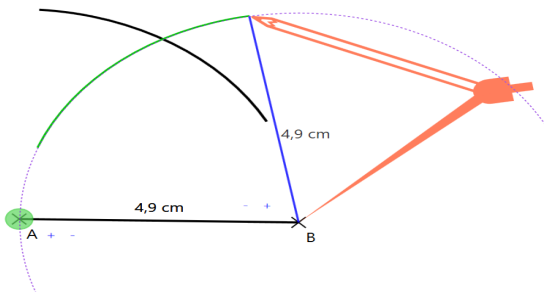
ABC et BCD (*cf.* la solution de l'enseignant dans le bloc 1 de la figure 8.1). Après avoir fini sa production, l'enseignant demande au tuteur de créer l'exercice.

Dans un premier temps, le module de génération de problème construit les noeuds du graphe, correspondant aux entités géométriques présentes dans le document. Les contraintes *Ref*, intrinsèques à ces noeuds, sont ajoutées (*e.g.* longueur(AB,2.5cm)).

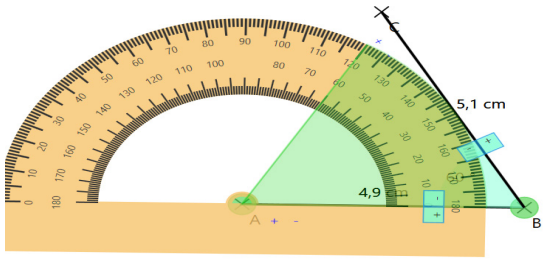
Dans un deuxième temps, les dépendances logiques sont ajoutées (arcs rouges en pointillés dans le bloc 3 de la figure 8.1. Ces relations de dépendances découlent de la modélisation du formalisme grammatical GMC-HPC de la *connaissance a priori*. En effet, un triangle est défini grammaticalement par ses constituants (si un segment est supprimé, le triangle l'est aussi). Ces relations de dépendances permettent en fait au module de propager les contraintes *Def* d'un noeud à ses voisins, et de créer de nouvelles contraintes sémantiques entre les noeuds (*e.g.* ABC rectangle en B $\implies$ [AB] $\perp$ [BC]). Ces contraintes sémantiques mettront à jour l'ensemble *Def* des noeuds concernés. Au delà de la création de nouvelles relations, cette propagation des contraintes de définition du concept peut aussi en supprimer. Notons bien que le seul noeud qui n'a pas de contraintes de longueur est le noeud AC dans la figure 8.1. Cela est dû au fait que c'est l'hypoténuse du triangle

ABC, et que par conséquent, il n'y a pas lieu d'avoir l'information sur la longueur (il existe bien sûr le cas où la présence d'un angle -*e.g.* $B\widehat{A}C$ ou $B\widehat{C}A$- rend cette information pertinente).

Dans un troisième temps, les contraintes structurelles *Struct*, telles que l'adjacence, l'intersection, qui ne dépendent pas d'une définition particulière d'un concept géométrique, sont extraites du moteur de reconnaissance 2D, et l'ensemble *Struct* est mis à jour pour chaque noeud concerné. Ces contraintes modélisées dans le graphe représentent toutes les conditions nécessaires et suffisantes à la résolution de l'exercice. Par cette modélisation, comme nous l'avons indiqué auparavant, nous nous affranchissons de la procédure suivie par l'enseignant pour évaluer la production de l'élève. L'ordre des tracés, et la façon de résoudre une étape (telle que l'utilisation du compas, ou du rapporteur pour dessiner un triangle équilatéral, *cf.* figure 8.2) importent peu. L'important est que l'action de l'élève ne viole pas les contraintes spécifiques au problème.



a) Compas et intersection de deux arcs



b) Rapporteur et angle de 60 degrés

FIGURE 8.2 – Deux stratégies pour la résolution d'un même problème

Les contraintes représentées dans le graphe de connaissance modélisent la connaissance spécifique au problème généré par le module. Nous pouvons parler ici de *connaissance déclarative*, qui correspond au savoir relatif à une notion : *Pour que ABC soit un triangle rectangle en B, il faut que [AB] soit perpendiculaire à [BC].*

Une fois toutes les contraintes du problèmes extraites et le graphe de connaissance construit, la génération de l'instruction textuelle est simple (*cf.* exemple dans le bloc 2 de la figure 8.1). Le module de génération de problème extrait les contraintes nécessaires et suffisantes pour la réalisation de chaque figure. Ce processus dépend de la figure en question et du concept géométrique associé. Par exemple, pour un triangle rectangle, la longueur des deux cotés adjacents à l'hypoténuse associée à l'indication que le triangle doit être rectangle sont des informations textuelles suffisantes pour le construire. L'instruction

ainsi extraite est affichée à l'enseignant qui peut la modifier si il le souhaite. La consigne est donc textuelle, mais notre système permet qu'elle prenne une forme d'image à reproduire, l'enseignant pouvant inclure à l'instruction une image d'une figure abstraite, avec comme consigne textuelle *"Reproduis cette figure"*. De plus, l'exercice peut consister à compléter une figure partiellement dessinée. Dans ce cas, l'enseignant spécifie dans l'interface les tracés de son exemple de solution à afficher à l'élève lors de sa résolution de l'exercice. Il pourrait par exemple décider que le segment [BC] soit déjà résolu et affiché à l'élève qui devra compléter la figure pour construire les deux triangles ABC et BCD.

Notons aussi qu'il est aussi envisageable que l'enseignant puisse ajouter des contraintes procédurales : par exemple l'obligation d'utiliser certain(s) outil(s), ou l'interdiction d'en utiliser d'autre(s). Le tuteur intelligent vérifiera alors que les contraintes choisies par l'enseignant sont valides (*i.e.* il n'est pas possible de construire un triangle équilatéral sans compas ni rapporteur) avant de les intégrer dans le modèle du problème. Nous ne développerons pas plus cet aspect dans ce manuscrit, précisons juste que cela fait partie des travaux en cours et des perspectives futures de travail.

8.3 Modélisation de l'état de résolution de l'élève

Après avoir présenté le module de génération de problèmes en nous focalisant sur la construction du graphe de connaissance modélisant la connaissance spécifique à un problème de géométrie, nous nous intéressons maintenant au module de l'apprenant et à la supervision en temps-réel de la résolution d'exercices. Le rôle du module de l'apprenant est d'interpréter sémantiquement les actions de l'élève mais aussi de les évaluer relativement à l'exercice. Au delà de représenter la connaissance spécifique au problème, le graphe de connaissance permet aussi de modéliser l'état de résolution de l'élève. Pour cela, chaque tracé reconnu par le moteur de reconnaissance et transformé en un nouvel élément géométrique, est donné en entrée au module de l'apprenant qui va rechercher dans le graphe de connaissance le nœud qui correspond (le mieux) à ce tracé. Deux cas sont possibles concernant cette recherche, soit l'élève labellise explicitement le tracé, c'est alors le nœud contenant ce label qui est recherché, soit il ne le labellise pas et la recherche s'appuiera sur les contraintes. Ces deux cas sont détaillés dans les sections suivantes.

8.3.1 Cas de base : vérification d'élément labellisé par l'élève

Notre système offre la possibilité à l'élève de labelliser ses figures. La figure 8.3 illustre ce processus. L'élève dessine un segment, et nomme ses extrémités A et B. Il a la possibilité de modifier ces labels en appuyant sur la lettre qu'il souhaite corriger, un menu apparaît pour lui permettre de réaliser cette modification. Dans ce cas, la mise en correspondance et l'évaluation de la pertinence de l'action est simple pour le module de l'apprenant. En effet, il va chercher dans le graphe de connaissance le noeud labellisé AB et remplacera son élément E par le segment tracé par l'élève. Une fois le noeud trouvé, le feedback correctif est généré et correspond aux contraintes satisfaites et celles violées dans le graphe de connaissance.

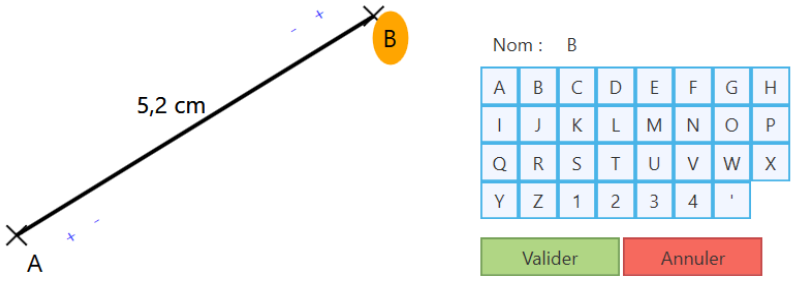


FIGURE 8.3 – Labellisation du segment par l'élève

Cela correspond bien au scénario traditionnel papier-crayon, où l'élève construit sa figure en nommant lui même ses points. Cependant, dans notre cas, nous voulons aller un cran plus loin pour fluidifier l'interaction homme machine en établissement une interprétation sémantique de l'action de l'élève, sans avoir besoin de labellisation de sa part.

8.3.2 Interprétation sémantique et mise en correspondance

Dans un contexte de dessin libre, l'élève est peu enclin à labelliser ses éléments au fur et à mesure qu'il les dessine. Il est plus probable qu'il trace la figure complète sans nommer les extrémités des segments au fur et à mesure, et qu'il ajoute les labels a posteriori. Cela rend le processus de vérification pas à pas (décrit plus haut) impossible. L'objectif étant de générer du feedback correctif immédiat, alors le module de l'apprenant doit interpréter chaque action afin **d'identifier ce que l'élève a voulu dessiner**. Ainsi, au delà de l'interprétation de la forme du tracé, il est nécessaire de mettre en oeuvre

une interprétation *sémantique* de l'élément reconnu, relativement au problème modélisé par le graphe de connaissance. Cette interprétation sémantique ne peut pas se baser sur la recherche simple du nœud contenant l'élément labellisé comme le cas précédent, mais sur une mise en correspondance du tracé dessiné avec un des noeuds du graphe de connaissance. Il est donc nécessaire de faire de la vérification de contraintes, en particulier vérifier les propriétés du tracé selon les contraintes des noeuds du graphe. Pour cela nous définissons un processus *dynamique* d'évaluation des contraintes. À chaque nouvelle action de l'élève, le module de l'apprenant n'évalue dans un noeud que les relations avec ses voisins qui sont soit à l'état partiel, soit à l'état complet. Les états des voisins évoluant au fur et à mesure de la réalisation de l'élève, une ré-évaluation des contraintes du noeud concerné est nécessaire.

Pour illustrer ce point, reprenons l'exemple de l'exercice illustré dans la figure 8.1 (deux triangles adjacents avec ABC rectangle en B, AB=2.5cm, BC=4cm, et BCD isocèle en B et BD=4cm.). Si dans le contexte d'une zone de dessin encore vide, l'élève trace le segment [AB], seule la contrainte de longueur sera évaluée. Si ensuite, l'élève dessine [BC], les contraintes du noeud BC qui seront évaluées sont sa longueur, sa perprendicularité et son adjacence avec le noeud AB. Dans le même temps, les contraintes du noeud AB évoluent pour inclure les relations de perpendicularité et d'adjacence avec le noeud BC, et seront ré-évaluées.

A) Évaluation des contraintes

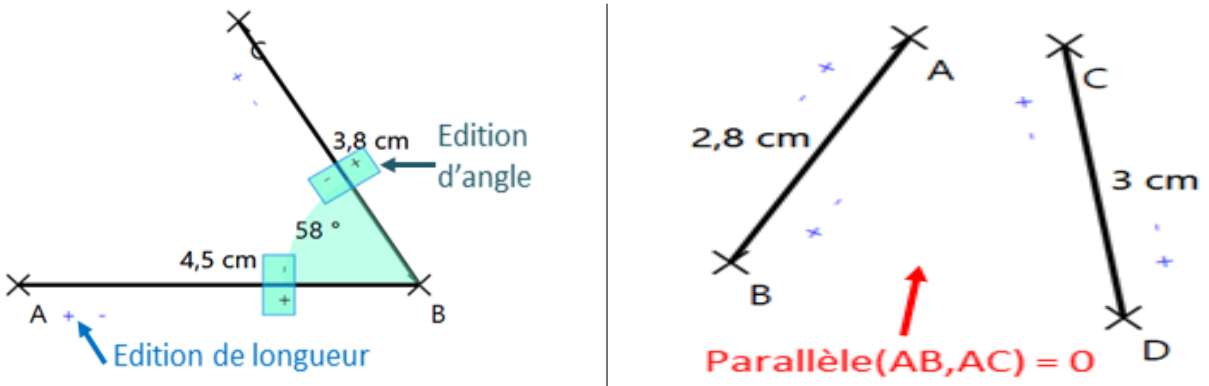
Nous distinguons deux types de contraintes :

— Les contraintes strictes : $\text{score}_{contrainte} \in \{0,1\}$;

— Les contraintes floues : $\text{score}_{contrainte} \in [0, 1]$.

Cette distinction découle intuitivement des pratiques de construction en géométrie, enrichies par les outils virtuels et le mode d'édition proposés dans IntuiGeo que nous détaillerons plus tard dans le manuscrit. Pour l'instant, précisons juste que l'application offre la possibilité d'éditer les tracés dessinés à main levée : *i.e.* modifier une longueur, ou la valeur de l'angle. Cette possibilité de modification implique que les contraintes relatives à ces deux propriétés sont considérées comme étant floues car il est possible de corriger l'erreur par une action d'édition (*cf.* figure 8.4.a).

A contrario, les contraintes telles que l'adjacence ou le parallélisme, etc. doivent être vérifiées strictement. En effet, il n'y a pas d'action possible pour résoudre une erreur d'adjacence ou de parallélisme autre que supprimer un segment et le re-dessiner (*cf.* figure



a) Édition possible, contrainte floue

b) Édition impossible, contrainte stricte

FIGURE 8.4 – Évaluation des contraintes selon les possibilités de dessin

8.4.b).

Plus spécifiquement, nous définissons le calcul du score pour la contrainte floue de longueur (et de même pour l'angle) comme suit :

$$Longueur(segment,\ valeurCible) = \frac{1}{1 + |longueurSegment - valeurCible|}$$

L'autre intérêt des contraintes floues est de permettre d'interpréter l'action de l'élève relativement au graphe de connaissance. Par exemple, pour l'exercice de la figure 8.1, si l'élève commence par dessiner un segment de 2.3 cm, on saura que le plus probable est qu'il ait tracé [AB] qui doit être de longueur 2.5 cm.

Ainsi, nous avons défini un processus de décision pour que le tracé de l'élève soit mis en correspondance, au fil de l'eau, avec le noeud du graphe le plus adéquat.

B) Prise de décision

Pour trouver le nœud correspondant le mieux à un tracé, nous avons défini un score d'adéquation entre chaque nouvel élément E_{new} reconnu et chacun des noeuds du graphe. La mise en correspondance suit le principe de maximisation du score. Nous définissons le score d'adéquation comme suit :

Definition 8.3.1. Adeq(E_{new}, n)=

$$\begin{cases} 0, & \text{si classe}(E_n) \neq \text{classe}(E_{new}) \\ 0, & \text{si } \text{État}_n = \text{complet} \\ 0, & \text{si } |\text{Struct}(\text{E}_{new}, n)| = 0 \\ \frac{\text{Struct}(E_{new}, n)+\text{Sem}(E_{new}, n)}{|\text{Struct}(n)|+|\text{Sem}(n)|}, & \text{sinon} \end{cases}$$

avec *Sem(n)* l'ensemble des contraintes sémantiques du noeud n, *Struct(n)* l'ensemble des contraintes structurelles de n, et *Struct(E_{new}, n)* la somme des scores des contraintes structurelles calculées en remplaçant E_n par E_{new}.

Dans le cas où le noeud n est à l'état partiel, *i.e.* déjà mis en correspondance avec un autre tracé de l'élève, il est considéré comme hypothèse valide pour E_{new} si et seulement si Adeq(E_{new}, n) > Adeq(E_n, n). Pour résumer, le noeud correspondant à un nouvel élément E_{new} tracé par l'élève est $n_{E_{new}}$ tel que : $n_{E_{new}} = \max_{n \in graphe} Adeq(E_{new}, n)$.

La figure 8.5 présente une étape de la résolution de l'exercice illustré dans la figure 8.1. Supposons que le noeud BC ait déjà été résolu par l'élève, étant donné que la seule contrainte à satisfaire à ce moment de la résolution était sa longueur de 4 cm.

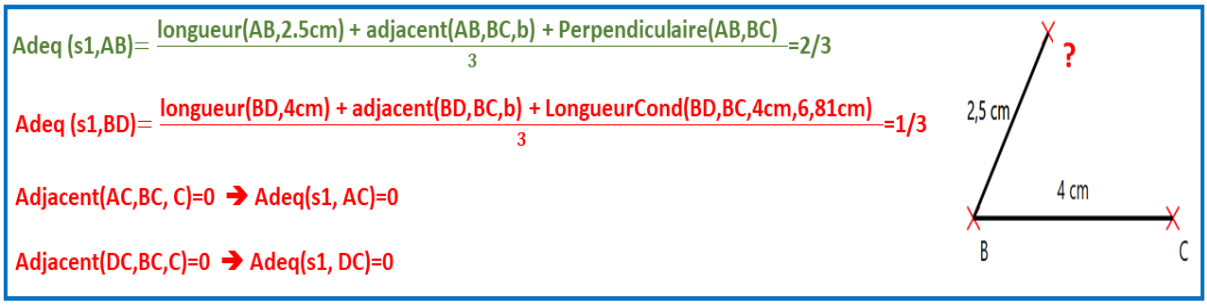


FIGURE 8.5 – Exemple de mise en correspondance

L'élève trace un nouveau segment que nous nommerons s1. Nous pouvons voir dans la figure 8.5 la présence d'une extrémité inconnue (dénotée par un point d'interrogation). Cela signifie que s1 n'a pas encore été mis en correspondance. Le module de l'apprenant mettra en compétition quatre hypothèses valides (AB, AC, BD, CD). Quand ce module évalue l'adéquation de s1 avec un noeud, il met en correspondance le label de l'extrémité inconnue avec l'élément E du noeud (*e.g.* ? est labellisé A quand le score *Adeq(s1, AB)* est évalué). Le noeud ayant le meilleur score d'adéquation verra son élément E remplacé

par le segment s1. Dans cet exemple, le noeud qui maximise l'adéquation est AB, puisque s1 satisfait deux contraintes sur trois (la contrainte [AB] $\perp$ [AC] n'est pas satisfaite). Une fois la correspondance établie, l'élément E du noeud est remplacé par E_{new}, le graphe de connaissance est mis à jour, et le feedback correctif est généré à l'élève. Notons donc que le graphe permet non seulement de représenter les contraintes spécifiques au problème, mais aussi de **représenter l'état de résolution de l'élève**. Cette interprétation sémantique de l'action de l'élève permet une interaction utilisateur orientée stylet fluide.

Via de calcul d'adéquation et la mise en correspondance, le module de l'apprenant peut 'suivre' la résolution du problème par l'élève. Cependant, certains dessins de l'élève peuvent engendrer une indécision dans la mise en correspondance. On parle alors, selon le cas, de rejet ou d'ambiguïté.

C) Cas de rejet et d'ambiguïté

Si aucun des scores d'adéquation n'est supérieur à 0, on est dans un cas où l'élément dessiné par l'élève n'a pas été "compris" par le module de l'apprenant. Nous mettons le mot entre guillemets, pour dire que ce n'est pas une erreur d'interprétation du module. Cela correspond plutôt à une action de l'élève qui n'est pas en relation avec le problème (par exemple le dessin d'un cercle dans un exercice sur les losanges). Tel est le cas du segment [GF] illustré dans la figure 8.6. Aucune contrainte structurelle n'étant satisfaite pour le mettre en correspondance avec AC, BD, ou AD (noeuds du graphe de la figure 8.1), [GF] est considéré comme un cas de rejet, *i.e.* il n'a pas d'impact sur l'état du graphe de connaissance et est ignoré par le module de l'apprenant.

Dans le cas de l'ambiguïté, il est possible que deux noeuds ou plus aient le même score d'adéquation pour un nouvel élément. Par exemple, ce pourrait être le cas entre les segments [BC] et [BD] qui ont la même contrainte de longueur de 4 cm. Dans ce cas, **et seulement dans ce cas**, nous nous référons à la procédure suivie par l'enseignant, et notamment l'ordre des tracés dessinés, pour choisir entre les deux hypothèses. Cette information est présente dans l'élément E des noeuds, qui correspond initialement au tracé de l'enseignant. Ce choix peut paraître arbitraire, cependant il n'a qu'un impact faible l'élève ayant la possibilité de valider implicitement cette interprétation ou de la rejeter en modifiant les labels automatiquement produits et associés aux extrémités du segment. La possibilité de solliciter l'utilisateur (*i.e.* lui demander de labelliser les extrémités) pour aider explicitement le processus de mise en correspondance en cas d'ambiguïté est envisageable pour des travaux futurs en veillant à ne pas le sur-solliciter.

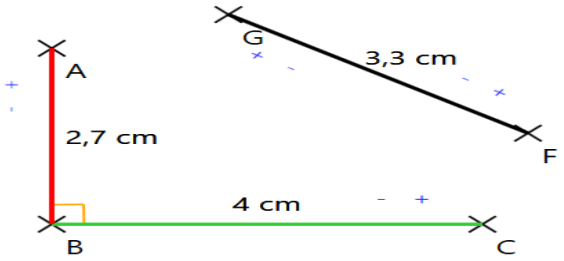


FIGURE 8.6 – Rejet de l'action de l'élève : le segment GF est considéré comme en dehors du plan de résolution, tous les scores d'adéquation dans le graphe sont nuls

8.4 Bilan

Dans ce chapitre, nous avons présenté notre approche pour la modélisation de la connaissance spécifique à un problème de construction en géométrie. Nous la représentons par un graphe de connaissance, qui contient les informations sur les concepts entrant en jeu dans l'exemple solution dessiné par l'enseignant, *i.e.* les éléments géométriques et leurs relations. En effet, le module de génération de problèmes, responsable de la construction du graphe, permet à l'enseignant de créer simplement et intuitivement un nouvel exercice (mode auteur). Ce graphe de connaissance contiendra toutes les contraintes nécessaires et suffisantes à la résolution du problème, quelque soit la stratégie choisie par l'élève dans sa réalisation.

Nous avons aussi vu comment le module de l'apprenant était capable d'évaluer, en temps-réel, les actions de l'élève, en les mettant en correspondance avec le graphe de connaissance. Cette interprétation sémantique permet de générer un feedback correctif qui décrit les erreurs de l'élève. Ce feebdack peut prendre la forme de couleur (*cf.* figure 8.6), ou une forme textuelle, ces modalités seront détaillées dans le chapitre 11.

Nous avons vu dans le chapitre 3 - section 3.2, l'importance du guidage dans les systèmes tutoriels intelligents à base de règles. Cette possibilité de guider l'élève, soit en avant pour les prochaines étapes, soit en arrière pour corriger d'éventuelles erreurs, est primordiale d'un point de vue pédagogique. Dans le chapitre suivant, nous présentons notre **module expert**, basé sur des techniques de planification, conçu pour synthétiser des stratégies de guidage à partir de l'état de résolution de l'élève.

Chapitre 9

Module expert et synthèse de stratégies de résolution

Dans le chapitre précédent, nous nous sommes intéressés à la connaissance déclarative spécifique à un problème de construction en géométrie, par la modélisation des contraintes d'un problème en graphe de connaissance et l'évaluation de la production de l'élève. Nous nous intéressons maintenant à la notion de *connaissance procédurale*, qui est relative au savoir-faire : *pour dessiner [AB] perpendiculaire à [AC], il faut utiliser l'équerre de telle façon*. En effet, pour être capable de guider l'élève dans sa réalisation, il faut que le tuteur ait la connaissance nécessaire pour pouvoir résoudre automatiquement des problèmes de construction. Cette connaissance procédurale est modélisée par le module expert, *via* la définition d'un domaine de planification dynamique avec des actions de dessin basées "règles-compas" qui modélisent le comportement ou les aptitudes de l'expert, *i.e* son savoir faire. Une des originalités de notre approche réside dans le fait que les actions de l'expert, au contraire des actions classiques de planification, sont définies de telle sorte qu'elles ont un impact sur le document. En effet, dans la définition d'une telle action, on modélise le tracé résultant de son application. Le module expert communique donc avec le module de reconnaissance 2D pour produire des actions de dessin, similaires à celles d'un élève ou d'un enseignant. Cette combinaison entre planification et reconnaissance de formes permet de produire des plans de correction adaptés à la grande variabilité des stratégies de résolution des élèves. Nous commençons donc dans ce chapitre par la définition du domaine de planification. Nous nous intéresserons ensuite aux différentes actions possibles qui modélisent la connaissance du module et à l'algorithme de planification qui permet de synthétiser des stratégies de résolution afin de générer des feedbacks de guidage adaptés à l'avancement de l'élève.

9.1 Le domaine de planification du module expert

Nous avons reformulé la résolution d'un problème de construction en un problème de planification. Le but ici est de générer une séquence d'actions de dessin afin de produire la figure cible. Notre approche consiste à décomposer la résolution de ce problème en un ensemble de sous-problèmes, en considérant la résolution d'un noeud du graphe de connaissance comme sous-problème de planification. Les actions de l'élève, interprétées par le module de l'apprenant, sont considérées comme des actions exogènes par le module expert, ce qui rend son domaine de planification dynamique. Nous nous plaçons dans le contexte de la planification en espace d'états [GNT04], où l'environnement est modélisé par les valeurs des variables d'états. Dans notre domaine, les variables d'états sont les contraintes géométriques explicitées dans la section 8.3.2. Dans ce domaine, une assignation de valeur v à une contrainte c est notée sous la forme c $\leftarrow$ v, telle que la contrainte c $\in$ C_{graphe}, l'ensemble de toutes les contraintes du graphe de connaissance, et v $\in$ [0, 1]. De ceci découle la description de l'état de l'environnement de planification, S $=\{$(c $\leftarrow$ v) | c $\in$ $C_{graphe}\}$. Par cette formulation, nous constatons que l'état de l'environnement représente bien l'état de résolution du problème.

Definition 9.1.1. Notre domaine de planification global est un système d'états transition $\sum$=(S, A, E, γ) avec :

- S $=\{$(c $\leftarrow$ v) | c $\in$ $C_{graphe}\}$: l'état de l'environnemt ;
- A : l'ensemble des actions de dessin, qui définissent le comportement de l'expert ;
- E : l'ensemble des évènements exogènes (actions de l'élève) ;
- γ=S $\times$ (A $\cup$ E) $\rightarrow$ 2^S la fonction d'état transition. Si a $\in$ A est applicable alors le système passe à l'état S'=γ(S,a).

9.2 Les actions de dessin du module expert

Dans la littérature [Koc16], une action a de planification, autrement appelée opérateur, est définie par :

$$\text{a} = (\text{Paramètres}_a, \text{Préconditions}_a, \text{Effets}_a)$$

où les paramètres$_a$ peuvent être soit des instances d'objets de l'environnement, soit des variables numériques. Les préconditions$_a$ sont un ensemble de contraintes qui doivent être satisfaites pour que l'action a soit applicable. Son application fait passer le système à l'état

s'= γ(s,a) = s $\cup$ Effets$_a$. Les effets$_a$ représentent de nouvelles assignations de valeurs à des contraintes du domaine.

Nous étendons cette définition classique des actions de planification, pour inclure les tracés manuscrits dans l'action. En effet, la planification ici pour le module expert ne consiste pas seulement à un trouver une séquence d'actions dont les effets permettront d'arriver à une solution. Le module expert construit son plan comme si c'était un utilisateur humain de l'application. Il est donc nécessaire de modéliser les possibilités de dessin manuscrit "artificiel" dans la définition des actions de planification. C'est pour cela que nous appelons ces actions des *actions de dessin*. Elles sont de type règle-compas, la base de la géométrie euclidienne, enrichies par la possibilité d'utiliser un rapporteur ou une équerre. Les actions d'édition sont aussi disponibles pour l'expert, comme elles le sont pour l'élève (voir figure 8.4.a). Considérant ces éléments, nous définissions donc les actions de dessin comme suit :

Definition 9.2.1. Les actions dans notre domaine de planification sont définies par un tuple a = (paramètres$_a$, préconditions$_a$, **tracés**$_a$, effets$_a$) avec :

- *paramètres*$_a$, des éléments géométriques ou des valeurs numériques impliqués dans l'action ;
- *tracés*, l'action(s) de dessin réalisée(s) par le module expert sur le document ;
- *préconditions*$_a$ et *effets*$_a$ un ensemble de contraintes (suivant les mêmes principes que ceux explicités plus haut dans la définition générale d'actions de planification$_a$).

La figure 9.1 illustre un exemple d'action de dessin qui vise à rendre deux segments perpendiculaires. La précondition à l'applicabilité de cette actions est que les segments en question soient existants dans le document (*DansDocument*), et adjacents. Le seul tracé associé à l'action, un signe orthogonal (*cf.* figure 9.2), est inspiré de la notion de codage en géométrie. **Le tracé de l'expert est interprété par le moteur de reconnaissance 2D, ce qui entraîne des transformations géométriques indiquées dans *effets*$_a$ dans le document**. La reconnaissance du tracé et, par conséquent, l'effet de l'application de l'action sont illustrés dans la figure 9.3.

Cette définition des actions, nous permet de représenter des actions de dessin complexes, telles que celles basées sur l'utilisation d'outils (compas, règle, ...). Par exemple, la figure 9.4 illustre un autre exemple d'action de dessin complexe qui consiste à savoir dessiner un segment s$_{new}$, en utilisant un compas. Les données nécessaires à la description de ce savoir-faire sont l'information sur sa longueur$_{s_{new}}$, une extrémité commune

($\text{ext}_{commune}$), avec un segment existant ($\text{s}_{existant}$), et sa distance avec l'autre extrémité de $\text{s}_{existant}$ (longueur_2).

Nous pouvons voir qu'il y a plusieurs tracés associés à cette action, illustrés dans la figure 9.5. En effet, il faut d'abord tracer deux arcs avec le compas (*ArcCompas*), sachant les longueurs citées plus haut. Enfin, il faut tracer le segment entre l'extrémité commune et l'intersection des deux arcs ($\text{intersection}_{Arcs}$).

À partir de cette modélisation des actions de dessin, la solution d'un problème de planification consiste à déterminer quelle action réaliser à partir d'un état du système. Pour cela, nous définissons deux métriques, qui vont permettre au module expert de

Action : RendrePerpendiculaire

Paramètres
(Segment : s_1, Segment : s_2)

Préconditions :
DansDocument(s_1) = 1 & DansDocument(s_2) = 1 & Adjacent(s_1, s_2) = 1 ;

Tracés :
TracéCodageOrthogonal(s_1,s_2) ;

Effets :
Perpendiculaire(s_1,s_2) = 1.

FIGURE 9.1 – Exemple d'action de dessin

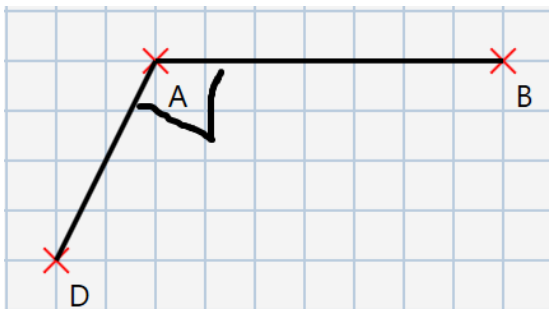


FIGURE 9.2 – Tracé : signe orthogonal

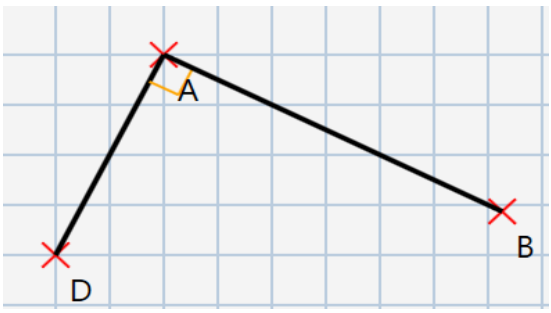


FIGURE 9.3 – Effet de l'action

Action : TracerSegmentAvecCompas

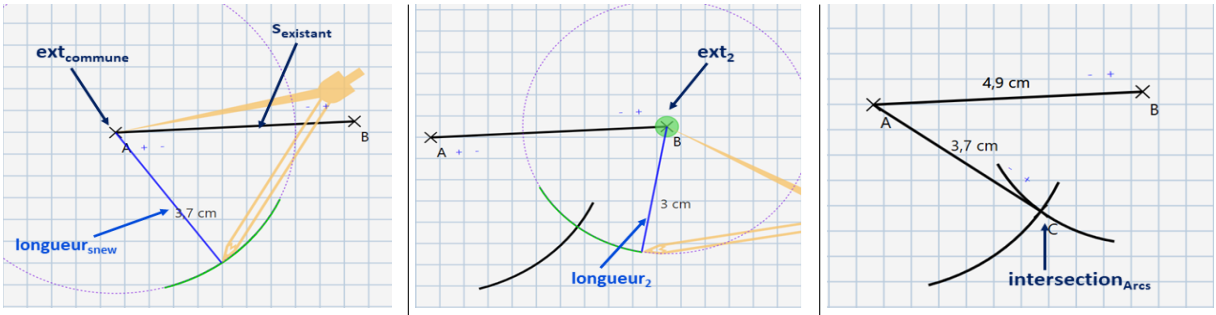
Paramètres
(Segment : $s_{existant}$, Segment : s_{new}, Extrémité : $ext_{commune}$, Extrémité : ext_2, Double : $longueur_{s_{new}}$, Double : longueur2)

Préconditions :
DansDocument(s_1) = 1 & DansDocument(s_{new}) = 0 ;

Tracés :
ArcCompas($ext_{commune}$, $longueur_{s_{new}}$) ;
ArcCompas(ext_2, $longueur_2$) ;
TracerSegment($ext_{commune}$, $intersection_{Arcs}$) ;

Effets :
DansDocument(s_{new}) = 1 & Longueur(s_{new}, $longueur_{s_{new}}$) = 1
& Distance($intersection_{Arcs}$, ext_2, $longueur_2$) = 1.

FIGURE 9.4 – Action de dessin avec plusieurs tracés



a) Premier tracé | b) Deuxième tracé | c) Troisième tracé

FIGURE 9.5 – Les trois tracés nécessaires à l'application de l'action *TracerSegmentAvec-Compas*

choisir quelle action appliquer à un moment donné. La première est un score, qu'il s'agira de maximiser, et qui est défini comme suit :

$$\text{Score(a)} = |\text{effets(a)} \cap \{\text{c, c} \in \text{C}_{graphe}\}|,$$

Ce score mesure la pertinence d'appliquer l'action a selon l'adéquation de ses effets avec les contraintes c à satisfaire dans le graphe de connaissance.

La deuxième métrique est une métrique de coût, qui permettra de distinguer entre deux actions qui ont le même score. Cette métrique de coût doit être minimisée, et est définie comme suit :

$$\text{Coût(a)} = |\text{Tracés(a)}|,$$

Défini ainsi, le coût se réfère donc à l'effort consenti pour réaliser une action : nous pouvons estimer par exemple qu'utiliser un rapporteur pour dessiner un angle de 60 degrés est moins coûteux qu'utiliser un compas. Dans le cas où deux actions ont le même score et le même coût, nous considérons que leur impact sur la résolution est équivalent, et l'expert applique l'une ou l'autre indifféremment.

À partir de ces définitions du domaine de planification, des actions, mais aussi des critères de choix parmi les actions possibles, voyons maintenant nous formalisons un problème de planification dans notre contexte de géométrie de construction.

9.3 Décomposition du problème de planification en sous-problèmes

D'après la définition de notre domaine, un problème de construction reformulé en un problème de planification est défini comme suit :

Definition 9.3.1. Le problème de construction est défini par $\text{P} = (\sum, \text{S}_0, \text{S}but)$ tel que

- $\sum$ le système d'états-transition définissant le domaine ;
- S_0 un état initial ;
- $\text{S}_{but} = \{(\text{c} \leftarrow 1) \mid \text{c} \in \text{C}_{graphe}\}$, un état final souhaité (la figure cible).

S_0 peut correspondre à une zone de dessin vide, où aucun noeud du graphe n'a été mis en correspondance avec une action de l'élève. S_0 peut par ailleurs se référer à n'importe quel état de résolution de l'élève. En effet, chaque nouvelle demande d'aide d'un élève correspond à un nouveau processus de planification de l'expert. S_{but} se réfère à l'état où

toutes les contraintes c du graphe de connaissance sont satisfaites, ce qui équivaut à dire que tous les noeuds sont résolus. Ainsi, la solution à ce problème est une séquence d'actions $<a_1, a_2, ..., a_n>$ qui mène à une séquence d'états $<s_1, s_2, ..., s_{but}>$, par l'application de la fonction d'état transition telle que $s_i = \gamma\ (s_{i-1}, a_i)$

Il est évident que pour résoudre toutes les contraintes du problème, il faut résoudre chaque noeud du graphe de connaissance un par un. La résolution d'un noeud, caractérisé par les états {*défaut, partiel, complet*}, est en elle-même un problème de planification, où l'état but consiste à ce que le noeud soit à l'état complet. Par conséquent, nous décomposons ce problème de planification en sous problèmes de planification P_{n_i} ($n_i \in$ graphe), chacun défini par :

Definition 9.3.2. Un sous problème P_{n_i} est un tuple $(\sum, n_i, S_{0_{n_i}}, S_{but_{n_i}})$

- $\sum$ le système d'états-transition définissant le domaine ;
- n_i le noeud appartenant au graphe de connaissance ;
- $S_{0_{n_i}}$ l'état initial du noeud ;
- $S_{but_{n_i}}$ est l'état cible, *i.e.* pour tout $c_{n_i} \in C_{n_i}$, $c_{n_i} \leftarrow 1 \iff n_i$. État = complet.

La résolution de ces sous-problèmes est plus simple que la résolution du problème global, puisque l'espace de recherche est moindre : on passe de l'ensemble des contraintes C_{graphe} qui doivent être satisfaites à l'ensemble des contraintes C_{n_i} qui doivent l'être pour chaque noeud n_i. Le plan global représentant la solution du problème est donc la séquence des sous plans $<p_{n_1}, p_{n_2}, ..., p_{n_m}>$, avec m le nombre de noeuds dans le graphe de connaissance.

9.4 Algorithme de planification

Après avoir défini le cadre du problème, nous nous intéressons maintenant au processus de planification à proprement parler. Comme nous l'avons expliqué, le principe est que le module expert essaie de résoudre un par un les sous-problèmes portés par les noeuds du graphe de connaissance. L'état du graphe de connaissance peut soit représenter une zone de dessin vide (tous les noeuds à l'état défaut), ou bien un état de résolution quelconque de l'élève. L'objectif et la méthode sont les mêmes pour les deux cas. Le processus est présenté dans l'algorithme 1. Le noeud à résoudre est choisi parmi ceux qui ne sont pas à l'état complet (ligne 3 dans l'algorithme). Cela déclenche un nouveau sous plan P_{noeud}. L'expert cherche ensuite l'ensemble des actions applicables sur le noeud choisi (ligne 6). Si

Algorithm 1 Algorithme de résolution

Entrée : S = état du graphe de connaissance, Goal = État but

1: Plan = { }
2: **Tant que** S != Goal **faire**
3: noeud = n ∈ graphe | (noeud.état = défaut ou noeud.état = partiel)
4: P_{noeud}= { }
5: **Tant que** ! noeud.état = complet **faire**
6: Actions= A_{noeud} | ∀ a ∈ A_{noeud}, a.applicable(noeud) est vrai
7: **si** Actions = ∅ **alors**
8: noeud =ChoixAutreNoeud(Graphe)
9: **si** noeud = ∅ **alors**
10: RetourEnArrière(S)
11: **fin si**
12: **sinon**
13: action= ChoixMeilleurScoreEtcoût(Actions)
14: S= γ(S, action)
15: P_{noeud} = P_{noeud} ∪ action
16: **fin si**
17: **fin Tant que**
18: Plan=Plan ∪ P_{noeud}
19: **fin Tant que**

l'ensemble est vide, le noeud n'est pas soluble à ce point du processus et l'expert retarde sa résolution en choisissant un autre noeud (lignes 7-8). Par exemple, il n'est pas possible de tracer l'hypoténuse [AC] d'un triangle ABC rectangle en B sans connaître sa longueur. Il faut évidemment tracer les autres cotés du triangle avant. S'il n'y a pas d'autre noeud soluble (ligne 9), S est considéré comme état non soluble, le module expert fait alors un retour en arrière à l'état S' précédent, en annulant la dernière action réalisée. Cette capacité de retour en arrière est nécessaire, puisqu'elle **simule la capacité d'un élève à supprimer une erreur si elle existe**. La figure 9.6 illustre un état de résolution non soluble de l'élève dans le contexte d'un exercice de construction de parallélogramme. Grâce à la capacité de retour en arrière dans l'algorithme de planification, **le module expert est capable de guider l'élève aussi bien en avant**, en lui indiquant les prochaines étapes à réaliser, **qu'en arrière**, en lui signifiant qu'il doit annuler sa dernière action pour pouvoir corriger ses erreurs.

Si, par contre, il existe des actions applicables pour le noeud choisi, la meilleure action est choisie parmi celles-ci en fonction des métriques de score et de coût (explicitées dans

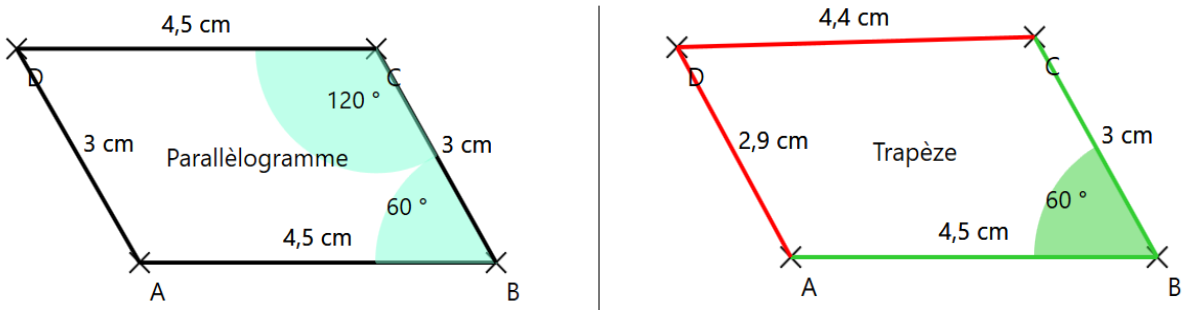


Figure cible

Figure de l'élève, état non soluble

FIGURE 9.6 – Intérêt du retour en arrière pour le guidage

la section 9.2). Ce processus est répété jusqu'à ce que le noeud soit à l'état complet, *i.e.* toutes ses contraintes sont satisfaites. Une fois le noeud résolu, d'autres sont sélectionnés pour être résolus jusqu'à ce que tous les noeuds du graphe soient à l'état complet, ce qui veut dire que l'état but Goal est atteint.

L'exécution d'un plan de résolution peut être représenté sous forme d'une arborescence, par exemple la figure 9.8 présente l'arbre d'exécution du plan solution complet de l'exercice présenté dans la section 8.2 dont la figure cible est rappelée ici dans la figure 9.7. L'arbre d'exécution montre bien l'avantage de décomposer la résolution en sous-problèmes de planification. En effet, l'espace de recherche est réduit au nombre d'actions applicables sur un noeud donné. Les actions de dessin relatives à la résolution des sous-problèmes de planification sont illustrées dans la figure 9.9. Le résultat du processus, la séquence d'actions de dessin (chemin en pointillés bleus sur la figure 9.8), est ensuite traduite à l'élève sous forme de feedback de guidage. Par exemple l'action de dessin *TraceSegmentAvecCompas(BD,BC,B,C,4cm,6.81cm)* est traduite textuellement à l'élève par "Trace les cercles respectivement de centre B et C et de rayon 4 cm et 6.81 cm pour construire le segment [BD]".

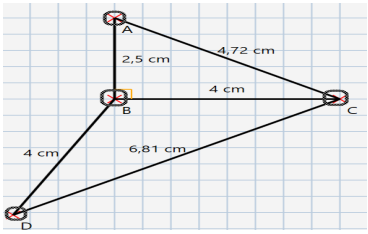


FIGURE 9.7 – Rappel de la figure cible dessinée par l'enseignant

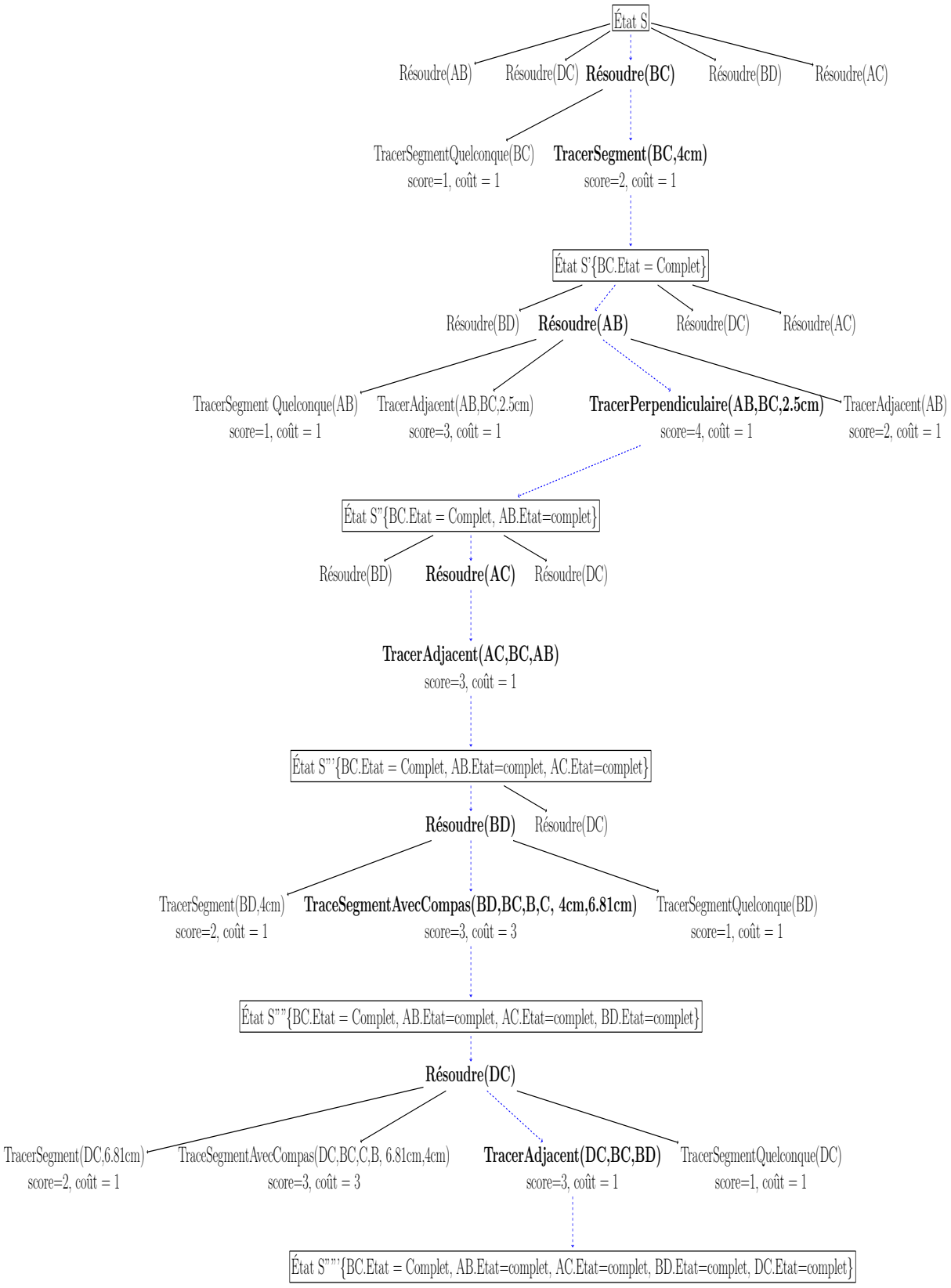
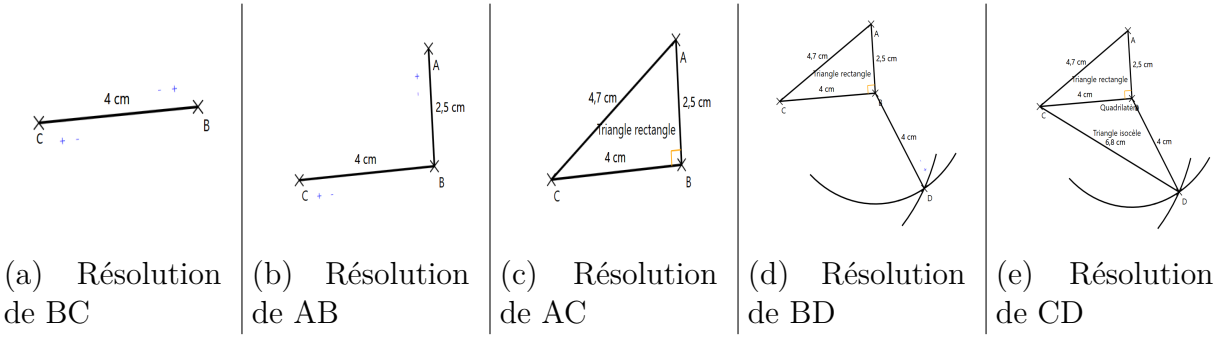


FIGURE 9.8 – Arbre d'exécution du plan solution



(a) Résolution de BC | (b) Résolution de AB | (c) Résolution de AC | (d) Résolution de BD | (e) Résolution de CD

FIGURE 9.9 – Actions de dessin relatives à l'exécution du plan de résolution

9.5 Bilan

Dans ce chapitre, nous avons présenté le module expert de notre moteur de supervision, responsable de la génération du feedback de guidage. Pour ce faire, nous définissons un domaine de planification dynamique, avec des actions de dessin expertes modélisant le savoir-faire de l'expert dans le domaine de la géométrie de construction. Par contraste aux techniques de planification classiques, nous intégrons l'impact réel de l'action (les tracés) dans sa définition. Cela permet de simuler le processus de construction de l'élève, et donc de générer des feedbacks précis et adaptés à son état de résolution. À partir de n'importe quel état de résolution, représenté par l'état du graphe de connaissance, le module expert est capable de générer un plan solution (séquence d'actions de dessin) menant à un état but où tous les noeuds du graphe sont résolus.

Plus généralement, nous avons présenté dans cette partie du manuscrit notre approche pour la conception d'un tuteur intelligent orienté stylet, par la combinaison du moteur de reconnaissance 2D et du moteur de supervision. Cette combinaison de techniques permet la génération de problèmes de construction à partir d'exemples solutions tracés par l'enseignant et reconnus par le formalisme grammatical GMC-HPC. Elle permet aussi l'interprétation sémantique des tracés de l'élève, au delà de la reconnaissance structurelle, par le module de l'apprenant. Enfin, comme nous l'avons vu dans ce chapitre, elle permet une définition originale du domaine de planification du module expert, modélisant le savoir-faire en géométrie de construction. Rappelons aussi que l'hybridation entre approche basée sur les règles (connaissance procédurale, module expert) et approche basée sur les contraintes (connaissance déclarative, graphe de connaissance) constitue une autre facette de l'originalité de notre moteur de supervision.

Dans le chapitre suivant, nous présentons les expérimentations relatives à l'évaluation

du moteur de supervision. Pour cela, nous évaluerons la capacité du module de l'apprenant à bien représenter l'état de résolution de l'élève ainsi que la capacité du module expert à synthétiser des stratégies de résolution à partir de n'importe quel état de résolution de l'apprenant.

Chapitre 10

Expérimentations et résultats relatifs à la performance du moteur de supervision

Dans ce chapitre, nous nous intéressons à l'évaluation du moteur de supervision, responsable de l'aspect tutoriel d'IntuiGeo. Plus précisément, nous évaluons la performance du module de l'apprenant, et du module expert. Les deux questions qui se posent sont les suivantes :

— Est-ce que le module de l'apprenant est capable d'analyser et d'évaluer les actions de l'élève relativement à l'instruction ?

— Est-ce que le module expert est capable de synthétiser des plans solutions, à partir d'un état de résolution quelconque ?

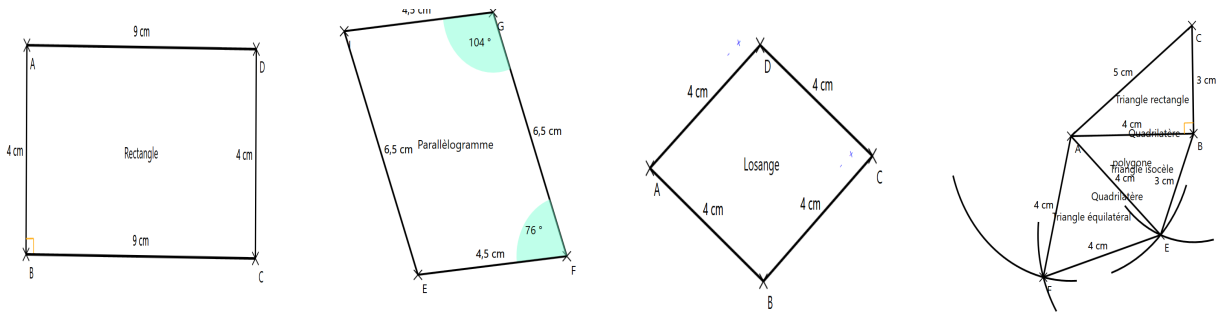
De la capacité du module de l'apprenant à évaluer les actions des élèves, c'est à dire de mettre en correspondance les éléments interprétés par le moteur de reconnaissance 2D avec les noeuds du graphe de connaissance, découle la pertinence du feedback correctif généré en temps-réel. De la capacité du module expert à trouver des plans solutions à partir de n'importe quel état de résolution de l'élève découle la possibilité de générer des feedbacks de guidage pertinents, comprenant des actions rétroactives, des retours en arrière, et des actions de dessin expertes qui représentent la connaissance procédurale du domaine.

10.1 Protocole d'évaluation

Pour répondre à ces deux questions, nous avons récupéré les données d'une expérimentation réalisée en Juin 2019 à laquelle ont participé 54 élèves de deux collèges bretons (collège La Roche aux Fées, Retiers, 35, et collège Paul Sébillot, Matignon, 22)). Nous les remercions ici de leur participation. Nous évaluons en batch la performance du moteur

de supervision en rejouant chaque fichier de réalisation de l'élève qui contient toutes ses actions (tracés manuscrits, utilisation des outils, suppressions, actions d'annulation, etc). Les 54 élèves ont réalisé quatre exercices, de difficulté variée, dont les consignes sont les suivantes :

1. *Trace un rectangle ABCD tel que AB = 4cm et BC = 9cm, cf. figure 10.1.a)* ;
2. *Trace un parallélogramme EFGH tel que EF = 4.5cm, FG = 6.5cm, et EFG = 76°, cf. figure 10.1.b)* ;
3. *Trace un losange ABCD tel que AB = 4cm et AC = 7cm, cf. figure 10.1.c)* ;
4. *Trace trois triangles tels que ABC est rectangle, ABD isocèle, et ADE équilatéral, cf. figure 10.1.d).*



a) exercice 1 b) exercice 2 c) exercice 3 d) exercice 4

FIGURE 10.1 – Solutions des quatre exercices proposés

10.2 Évaluation de la performance du module de l'apprenant

La table 10.1 illustre la capacité du module de l'apprenant à évaluer les réalisations des élèves pour l'ensemble de test. Le but de l'expérimentation est de vérifier si le module est capable de représenter correctement l'état de résolution de l'élève, *i.e.* si la mise en correspondance des tracés avec les noeuds du graphe de connaissance est pertinente. Le terme *nbProd* correspond au nombre total de productions pour chaque exercice. Le terme *succès* dans le tableau signifie que le module estime que l'élève a réussi l'exercice (réalisé la figure complète), et le contraire pour le terme *échec*. *FN* se réfère aux faux négatifs, des productions qui sont correctes mais que le module a évalué comme erronées.

Nous pouvons remarquer le taux de précision (Pre) est à 100 %, ce qui veut dire que le tuteur ne prend pas de risque quand il évalue la production de l'élève. Cependant, il y a deux cas de faux négatifs (1 pour l'exercice parallélogramme, 1 pour l'exercice losange). Les élèves ont adopté des stratégies qui ont fait que le module de l'apprenant n'a pas été capable de bien mettre en correspondance les tracés dessinés avec les noeuds du graphe. La solution de ce problème réside dans l'enrichissement de la base des contraintes qui modélise la connaissance spécifique au domaine, notamment les contraintes *Def* qui représentent les théorèmes et les axiomes de géométrie (typiquement, pour le faux négatif du losange, inclure les diagonales et leurs propriétés dans le graphe de connaissance, même si elles n'ont pas été dessinées par l'enseignant, pour pouvoir interpréter la stratégie de construction non reconnue). On obtient au final une justesse (*Acc*uracy) de 99.03% et un rappel de 97.72% sur un ensemble statistiquement significatif, ce qui nous amène à considérer la performance du module comme bonne.

Table 10.1 – Performance du module de l'apprenant

Problème	nbProd	Succès	Échec	TP	FN	Pre	Rap	Acc
Rectangle	75	55	20	55	0	100%	100%	100%
Parallélogramme	94	23	71	23	1	100%	95.65%	98.94%
Losange	24	5	19	5	1	100%	83.3%	95.8%
Triangles	13	3	10	3	0	100%	100%	100%
Total	206	86	120	86	2	100%	97.72%	99.03%

10.3 Performance du module expert

Comme nous l'avons explicité dans le chapitre 9, le module expert doit être capable de trouver des plans solutions partant de n'importe quel état de résolution de l'élève. Si il n'y a pas de solution possible, le module expert fait un retour en arrière jusqu'au dernier état soluble atteint par l'élève, et propose un plan solution à partir de ce dernier. Ce sont ces aptitudes qui vont permettre au module expert de générer un guidage pertinent et adapté à l'élève. La table 10.2 présente la performance de ce module sur chaque réalisation appartenant à l'ensemble des vrais négatifs précédents (nb échecs - faux négatifs). Nous établissons les critères d'évaluation suivants :

— lg Plan opt : la longueur du plan optimal trouvé par le module expert, à partir d'une zone de dessin vide (état initial) ;

— lg Plan corr : la longueur moyenne du plan de correction généré à partir de l'état de résolution de chaque élève (qui par la suite est traduit en feedback de guidage) ;
— nb retours en arrière : le nombre moyen de retours en arrière effectués sur chaque réalisation pour arriver à un état soluble ;
— Plans trouvés : le pourcentage de réussite dans la génération de plan pour chaque lot de réalisations.

TABLE 10.2 – Performance du module expert

Problème	Quantité	lg Plan optimal	lg plan correction	nb retours en arrières	Plans trouvés
Rectangle	20	6	4	5	100%
Parallélogramme	70	5	6.49	3.86	100%
Losange	18	5	3.94	3.55	100%
Triangle	10	8	6.9	13.9	100%

Nous remarquons que **le module expert a été capable de générer un plan correction dans 100 % des cas**. Cela démontre qu'il est toujours capable de trouver un plan solution, quelque soit l'état de résolution de l'élève, soit par l'application directe d'une séquence d'actions de dessin, soit en effectuant un retour en arrière jusqu'à arriver au dernier état soluble de l'élève. Un autre point intéressant est la différence entre la longueur moyenne des plans de correction et la longueur du plan optimal qui, couplée à l'analyse du nombre moyen de retours en arrière, démontre que le processus de planification est réellement adapté à l'état de l'apprenant, et que le module expert ne se contente pas de revenir en arrière jusqu'à l'état initial pour appliquer son plan optimal. Ce résultat est important, puisque nous avons besoin de générer des feedbacks de guidage réellement pertinents et adaptés à l'avancement de l'élève. Concernant le nombre élevé de retours en arrière pour l'exercice portant sur les triangles, cela s'explique par la difficulté de l'exercice (sur 13 élèves qui l'ont fait, seulement 3 ont réussi). Cette difficulté engendre dans certains cas un désintéressement et une volonté de dessiner librement des figures, d'où le nombre important de retours en arrière nécessaires pour arriver à un état soluble de l'élève.

10.4 Bilan

Dans ce chapitre, nous avons présenté les résultats d'évaluation du moteur de supervision sur des données d'exercices réalisés par 54 élèves. Le module de l'apprenant est capable de superviser et d'interpréter les actions de l'élève relativement à la consigne et au modèle de l'exercice représenté par le graphe de connaissance. Ce module est respon-

sable de la génération du feedback correctif en temps-réel. Le module expert est quant à lui capable de synthétiser des stratégies de résolution/correction à partir de n'importe quel état de résolution de l'élève. Cela permet de générer des pistes et des guidages pour les élèves qui sont bloqués. Ces résultats ont été publiés dans [1].

Dans la partie suivante, nous nous intéresserons au prototype d'IntuiGeo, en nous focalisant sur les outils virtuels réalistes d'une part, et sur la typologie des feedbacks de correction et de guidage d'autre part. Nous y présenterons aussi une étude en écosystème (*i.e.* en classe) de l'impact pédagogique d'IntuiGeo sur la performance des élèves.

Quatrième partie

Prototypage et expérimentation en classe

Chapitre 11

Prototypage d'IntuiGeo

L'implémentation de l'IHM d'IntuiGeo a été réalisée avec l'appui de plusieurs mois d'ingénieurs, nous profitons de cette introduction pour les remercier encore une fois. Techniquement parlant, le système a été réalisé sous .Net (pour tablettes Windows). Nous avons développé un prototype complet qui inclut le tuteur intelligent, ce qui nous a permis de :

— Faire une évaluation des moteurs de reconnaissance 2D et de supervision comme explicité dans les chapitres 6 et 10 ;
— Faire des tests d'usage dans les écoles avec le soutien du LP3C [1] et du Loustic [2] dans le cadre du projet e-Fran actif ;
— Commencer à mesurer les impacts pédagogiques d'IntuiGeo sur l'apprentissage des élèves, en réalisant des expérimentations en classe, toujours avec le soutien du LP3C/Loustic.

Dans ce chapitre, nous nous intéressons donc à la mise en oeuvre de notre système tutoriel intelligent orienté stylet InutuiGeo, et plus particulièrement à l'aspect interaction homme-machine.

L'interprétation à la volée des tracés manuscrits permet de simuler l'approche traditionnelle papier-crayon, mais il est nécessaire de proposer aussi une manipulation intuitive de l'outil pour rendre l'environnement numérique transparent à l'élève et lui permettre de se concentrer sur sa tâche. La précision du stylo sur le papier pouvant parfois être difficile à atteindre avec le stylet, nous proposons des fonctions d'éditions qui permettent d'atteindre la précision nécessaire aux problèmes de géométrie, ce qui n'est pas évident dans le contexte de dessin à main levée (édition de longueur, d'angle, de position de points...). Au delà de la possibilité de dessiner librement des figures géométriques, nous proposons des outils virtuels réalistes, dont le comportement simule celui des outils classiques. Nous nous intéresserons aussi dans ce chapitre aux différentes typologies de feedbacks générés par le tuteur : des feedbacks visuels, de correction et de guidage. Nous nous sommes

1. Laboratoire de Psychologie : Cognition, Comportement, Communication
2. Laboratoire d'observation des usages des TIC

basés sur un processus de Conception Centrée Utilisateur (CCU), en partenariat avec le LP3C, qui a permis l'évolution de l'outil en adéquation avec les besoins des enseignants partenaires du projet et des élèves qui ont pris part aux expérimentations successives en classe. Enfin, nous verrons ce que peut apporter notre système à l'enseignant, notamment la capacité de revoir la stratégie qu'a suivie l'élève pour résoudre l'exercice.

11.1 Interface et fonctions d'éditions

11.1.1 Interface

L'interface d'IntuiGeo est illustrée dans la figure 11.1. Elle est composée de quatre zones : la zone de dessin (rectangle vert), la zone de l'exercice (rectangle rouge), la zone de menu (rectangle jaune), et la zone de calque (rectangle gris).

Zone de dessin

La zone de dessin est la zone principale de l'application, où l'utilisateur (enseignant/élève) dessine librement ses figures géométriques avec le stylet et manipule des outils virtuels avec ses doigts.

Zone d'exercice

La zone d'exercice comporte l'instruction textuelle, une jauge de complétion représentant les étapes réalisées et celles comportant des erreurs, et un bouton de demande d'aide sur lequel l'élève appuie pour obtenir du guidage de la part du tuteur. Dans la figure 11.1, l'utilisateur a utilisé la règle pour dessiner deux segments [AB] et [BC] dans le contexte de la résolution d'un exercice de construction de losange. La coloriation de [BC] en vert signifie que sa longueur est valide en rapport à l'instruction, contrairement à celle de [AB].

Zone de menu

La zone de menu permet l'accès aux outils virtuels qu'il est possible d'utiliser (compas, rapporteur, règle, équerre) et à d'autres actions d'édition classiques en IHM (annuler, refaire, enregistrer...).

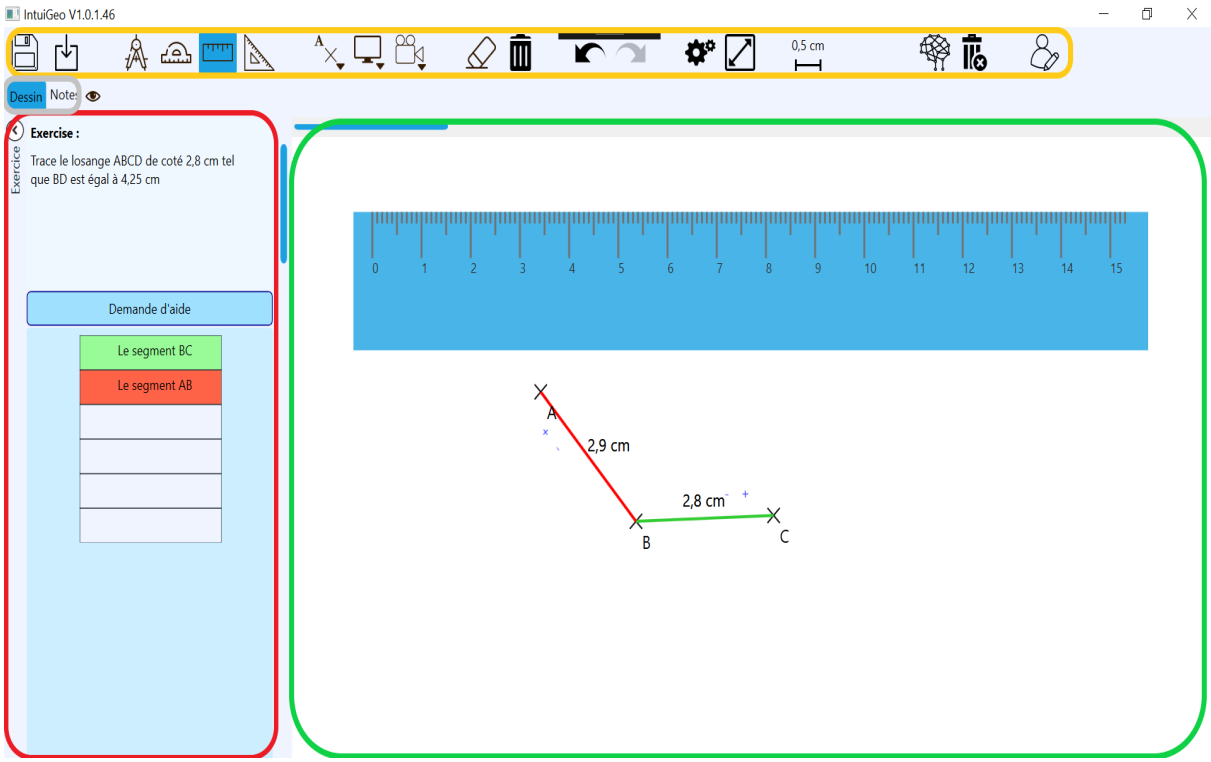


FIGURE 11.1 – Interface d'IntuiGeo : zone de dessin encadrée en vert, zone d'exercice encadrée en rouge, zone de menu encadrée en jaune, zone de calque encadrée en gris

Zone de calque

La zone de calque permet à l'utilisateur de choisir entre deux calques proposés par le système : un calque de dessin pour produire des figure géométriques, et un calque d'annotation pour ajouter des annotations sur la zone de dessin (utile notamment pour l'enseignant lors de la correction d'une production de l'élève).

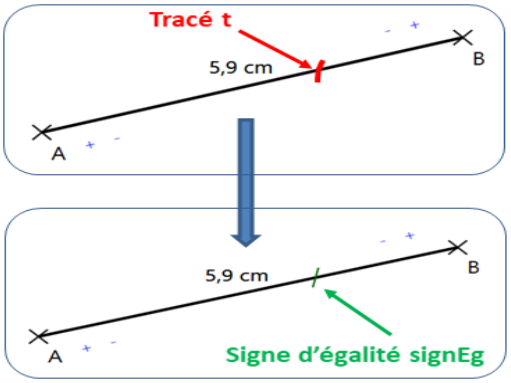
11.1.2 Fonctions d'éditions

Le but des fonctions d'éditions est d'offrir un moyen à l'utilisateur de modifier ses productions de façon intuitive. Certaines de ces fonctions d'édition sont formalisées directement dans la grammaire, tandis que d'autres le sont par des gestes d'édition de type glisser-déposer.

Formalisation du codage en géométrie

Dans la condition traditionnelle papier-crayon, nous utilisons souvent des symboles, ou codes, pour expliciter des propriétés géométriques telles que l'orthogonalité ou l'égalité. Nous étendons cette notion par la définition de gestes de commandes qui non seulement permettent d'expliciter ces propriétés, mais aussi de les appliquer sur les figures du document. Ces gestes sont intégrés dans des règles de production comme tracés à consommer pour la transformation d'éléments existants. Nous avons déjà vu le cas du geste d'orthogonalité dans la section 9.2. Nous nous intéressons ici au signe d'égalité entre deux segments, modélisé par deux règles de production dans les figures 11.2 et 11.3. Un tracé t est transformé en un signe d'égalité si il intersecte la zone $longueurSegment$ d'un segment s1 existant. Le segment [CD] est transformé pour que sa longueur soit égale à celle de [AB], si le tracé dessiné par l'utilisateur est un signe d'égalité, et qu'il en existe un autre appartenant au segment [AB] (*cf.* figure 11.3.b).

SigneEgalite : SigneEg → **Tracé t** avec :
Preconditions :
(Segment :s1)[LongueurSegment] (t)[Un]
Contraintes :
Reconnaisseur(t, signe)
Postconditions :



a) Règle de production b) Composition

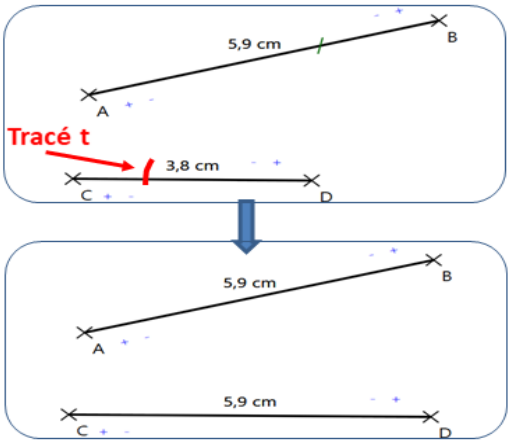
FIGURE 11.2 – Formalisation du signe d'égalité en GMC-HPC sur un segment

Fonctions d'édition de type glisser-déposer

Nous avons vu dans la section 8.3.2, figure 8.4, que l'utilisateur a la possibilité d'éditer la longueur d'un segment ou la valeur d'un angle, par l'utilisation de boutons + et - placés sur les extrémités des éléments géométriques. Chaque élément dans le document a des points d'ancrage ou de sélection. Nous laissons donc la possibilité à l'utilisateur de rallonger des segments, de modifier les côtés d'un angle, par un appui long du stylet sur ces points d'intérêt. Le même principe est pris en compte pour la modification libre de

Segment : SegModifié → segment s1, Tracé t avec :
Preconditions :
(s1)[LongueurSegment] (t)[Un] &
(Segment :S2)[longueurSegment](SigneEgalité : sig)[un]
Contraintes :
Reconnaisseur(t, signe)
Postconditions :
segment postconditions

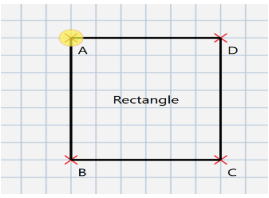
a) Règle de production



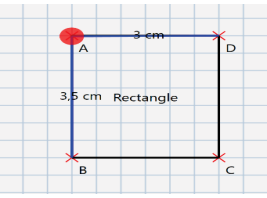
b) Composition

FIGURE 11.3 – Formalisation de l'égalité entre deux segments en GMC-HPC (le premier étant déjà identifié sur la figure 11.2)

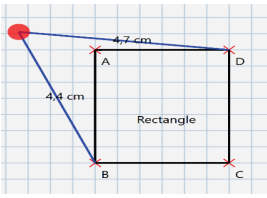
figures géométriques. Ce mode d'édition orienté stylet est illustré dans la figure 11.4. Le point d'ancrage sélectionnable est affiché en jaune sur la figure 11.4.a, après un appui long il s'affiche en rouge comme sur la figure 11.4.b. Puis, lors du mouvement du stylet (glissement du point), les éléments qui seraient modifiés sont présentés en bleu (*cf.* figure 11.4.c), enfin lorsque le déplacement est fini (levé du stylet ~ déposer), les éléments initiaux qui ont subi une modification sont supprimés du document, de nouveaux tracés artificiels sont générés à partir de la nouvelle position, pour ensuite reconnaître une nouvelle figure (quadrilatère, figure 11.4.d).



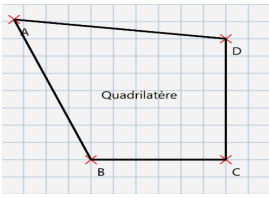
a) Choix du point d'ancrage



b) Appui long avec stylet



c) Modification de la position du point



d) Transformation de la figure

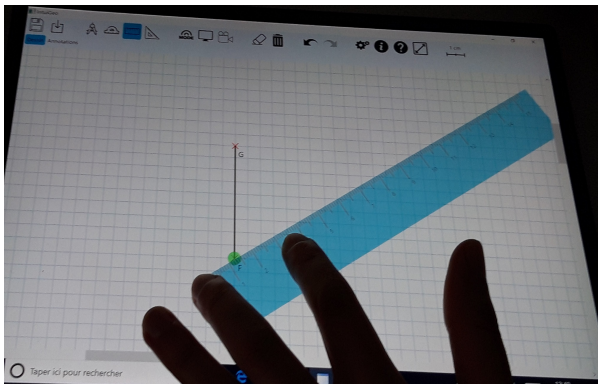
FIGURE 11.4 – Édition libre (orientée stylet) de figures géométriques par glisser-déposer

11.2 Outils virtuels et conception centrée utilisateur

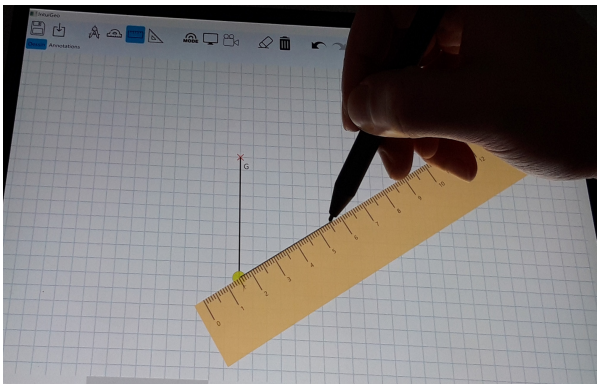
Nous proposons une interaction intuitive en ce qui concerne les outils virtuels, l'idée étant de simuler au maximum les outils classiques pour éviter une phase d'entraînement fastidieuse et garantir une transférabilité de l'apprentissage vers le papier/crayon. Par conséquent, ils sont manipulables avec les doigts et permettent de réaliser les mêmes actions qu'avec le format physique traditionnel utilisé sur papier. Le numérique permet toutefois d'apporter une aide supplémentaire, en profitant de la notion de point d'ancrage. Cette notion permet par exemple de fixer une règle sur une extrémité d'un segment pour dessiner un tracé (*cf.* figure 11.5). Au delà d'aider à la construction de figures, un outil peut aussi servir à vérifier des propriétés, cette vérification est facilitée avec l'utilisation d'un outil virtuel, celui-ci pouvant être affiché de couleur différente selon la validité de la propriété. Cette approche est mise en oeuvre par exemple avec l'équerre qui permet de vérifier que deux segments sont perpendiculaires. Ce processus correspond à faire changer l'outil de couleur lorsque la propriété d'orthogonalité est vérifiée, passage à la couleur verte tel qu'illustré dans la figure 11.6. Concernant l'utilisation d'un rapporteur pour tracer un angle, comme illustré sur la figure 11.7, l'utilisateur doit d'abord ancrer le rapporteur sur un segment existant, et ensuite choisir la valeur de l'angle en traçant une cote sur l'outil.

Comme nous l'avons indiqué, nous avons suivi un processus de conception centrée utilisateur. Le principe de ce processus est de prendre en compte le retour des utilisateurs lors de chaque phase de développement, afin que les évolutions du système soient en accord avec leur besoins. Notre partenariat avec le LP3C/Loustic de l'université Rennes 2 ainsi que la participation de collèges pilotes en Bretagne aux différentes phases de test ont permis une évolution itérative du logiciel, notamment pour le design et le choix des modalités de manipulation des outils virtuels.

La figure 11.8 illustre ce processus de conception, qui se caractérise par des cycles d'implémentation-évaluation, en montrant l'exemple de l'évolution de l'outil compas. Dans la première version du logiciel expérimentée par les élèves, le principe de l'utilisation du compas était le suivant : l'élève appuie avec le stylet sur un point d'accroche, puis choisit le rayon du compas, et dessine son arc, toujours avec le stylet. Les retours utilisateurs nous ont montré la difficulté d'utilisation découlant du fait que la manipulation de l'outil physique compas était assez éloignée du processus que nous proposions. C'est pour cela que dans la deuxième version du compas virtuel, nous avons changé le paradigme d'utilisation afin de simuler l'outil traditionnel. L'élève manipule le compas avec les doigts, pour

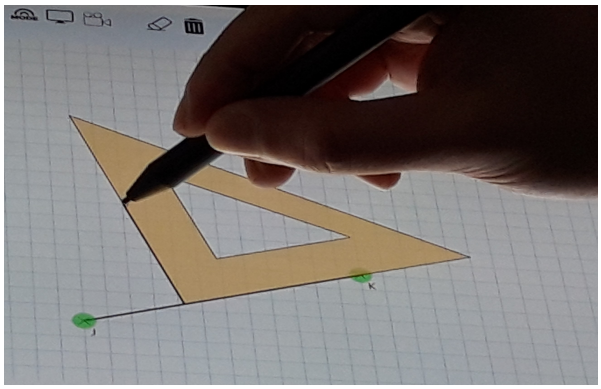


Manipulation avec les doigts et ancrage

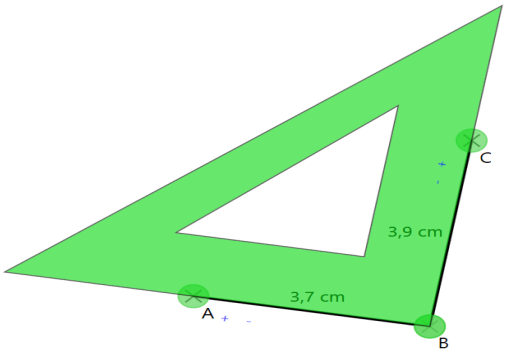


Dessin avec stylet

FIGURE 11.5 – Comportement de l'outil virtuel règle

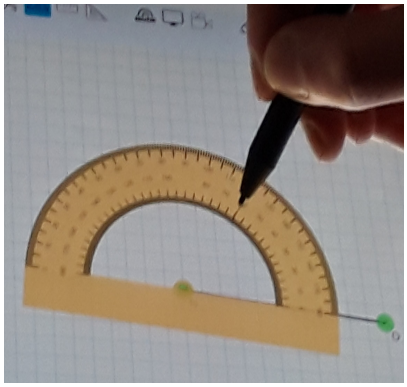


Utilisation pour dessin

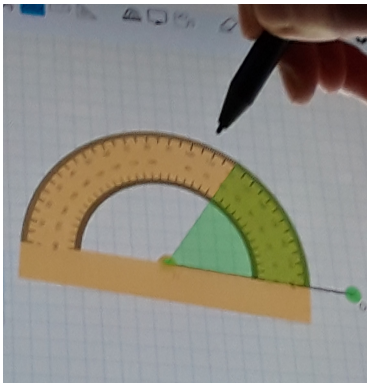


Utilisation pour vérification

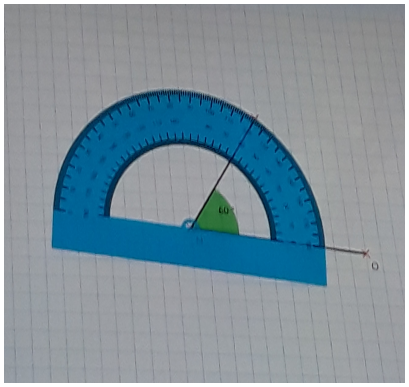
FIGURE 11.6 – Comportement de l'outil virtuel équerre



(a) Ancrage sur segment



(b) Choix de la valeur



(c) Angle reconnu

FIGURE 11.7 – Comportement de l'outil rapporteur

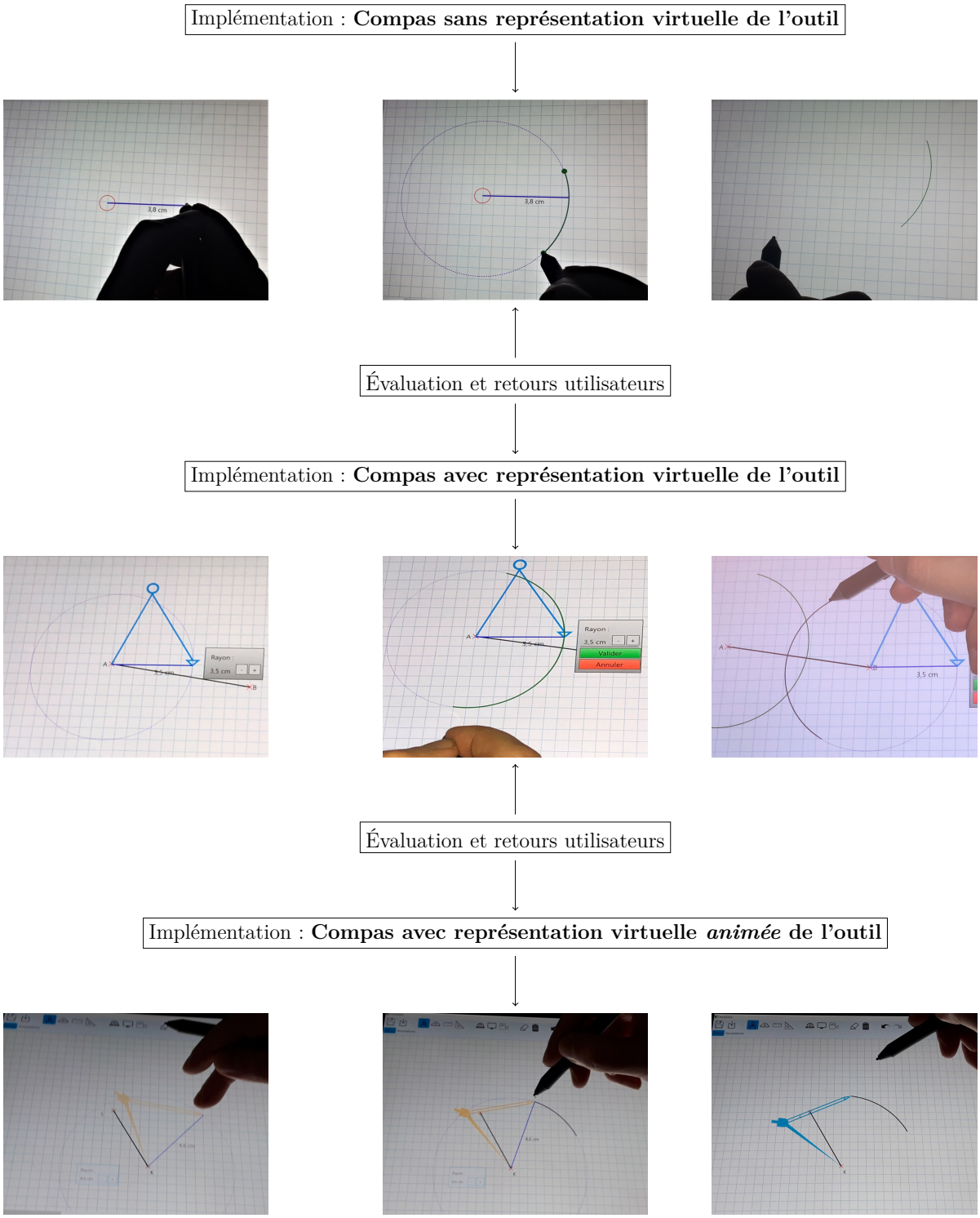


FIGURE 11.8 – Évolution de l'outil compas dans le cadre de la conception centrée utilisateur

le mouvoir dans la zone de dessin, placer la pointe sur un point d'accroche et changer le rayon, et ensuite dessiner l'arc ou le cercle avec le stylet. Les retours sur cette deuxième version étaient meilleurs, mais il avait été observé que le mouvement habituel du compas lorsque l'on trace un arc n'était pas reproduit. De fait, après un nouveau cycle de développement, nous avons proposé une dernière version de l'outil intégrant l'amélioration de son ergonomie, en augmentant la sensation de réalisme (*e.g.* le mouvement du compas suit le mouvement du stylet lors du traçage de l'arc) on peut alors parler de compas avec représentation virtuelle animée de l'outil.

11.3 Feedback : visuels, de correction, et de guidage

Nous avons présenté jusqu'ici l'ensemble des interactions proposées dans le sens utilisateur vers système. Il est aussi nécessaire que le système produise de l'information vers l'utilisateur, pour cela nous avons mis en oeuvre différents types de feedbacks que nous décrivons dans la section suivante.

11.3.1 Feedback visuels

Au travers l'ensemble des exemples présentés jusqu'alors, des feedbacks que nous qualifions de visuels sont déjà présents sur les figures dessinées à main levée. En effet, parmi ces feedbacks visuels nous pouvons citer la remise au propre d'un tracé qui permet d'informer l'utilisateur que le moteur/système a bien interprété son action. Nous pouvons aussi considérer l'affichage des longueurs des segments ou encore de la valeur d'un angle comme des feedbacks visuels.

Lors des premières démonstrations du système, nous avons eu un retour indiquant que ces feedbacks pouvaient être considérés comme trop facilitants dans la résolution de problème. Nous avons donc intégré un mode qui permet d'activer/désactiver l'affichage des longueurs et des angles, ainsi qu'un mode qui permet d'afficher les tracés bruts de l'utilisateur (non remis au propre). Ainsi ces feedback, bien que basiques, permettent une interaction fluide du système avec l'utilisateur dans le contexte du dessin à main levée de figures géométriques.

Intéressons nous maintenant aux deux types de feedbacks générés par le moteur de supervision : des feedbacks de correction générés à partir du module de l'apprenant, en se basant sur la connaissance déclarative modélisée par les contraintes du graphe de connaissance, et des feedbacks de guidage générés à partir du module expert, en se basant sur la connaissance procédurale modélisée par les règles de planification. Nous illustrons ces typologies des feedbacks par un exemple de résolution d'un exercice sur la tablette. La figure 11.9 présente un problème de reproduction de figure (IntuiGeo permet à l'enseignant d'ajouter une image à l'instruction).

11.3.2 Feedback de correction

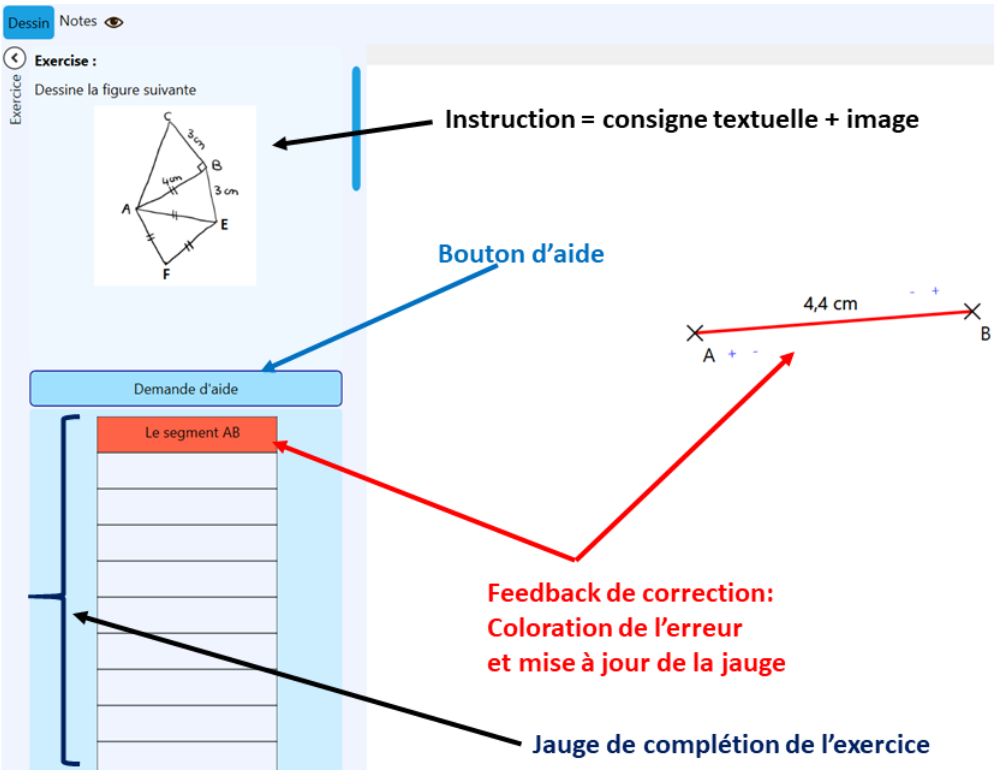


FIGURE 11.9 – Feedback de correction - version finale

Pour la résolution de l'exercice, supposons que l'élève dessine un premier segment tel qu'illustré sur la figure 11.9. Ce segment sera mis en correspondance avec le noeud AB du graphe de connaissance et la jauge de complétion (à gauche) se met par conséquent à jour indiquant avoir reconnu AB, mais se colorie en rouge pour signifier qu'il y a des erreurs dans ce noeud. De même le segment est colorié en rouge pour signifier qu'il s'agit

d'une erreur de longueur. Ces deux aspects représentent notre *feedback correctif*.

Notons que d'autres versions du logiciel ont comporté une autre forme de feedback de correction textuel plus explicite, avec une zone décrivant les erreurs en lieu et place de la jauge de complétion tel qu'illustré sur la figure 11.10.

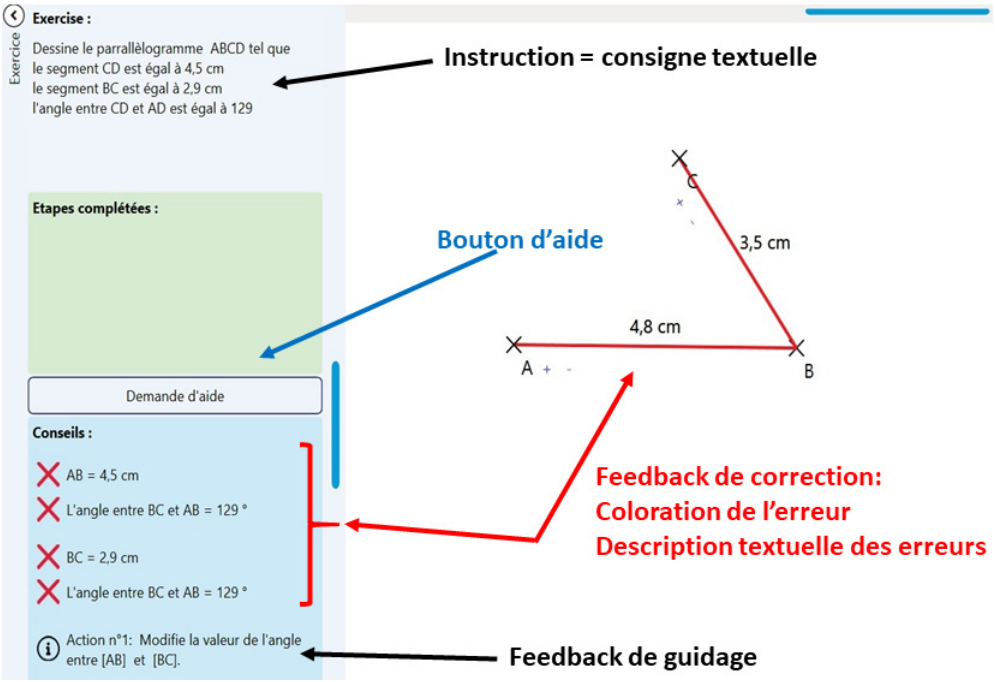


FIGURE 11.10 – Feedback textuel de correction - version initiale

Cependant, au cours des expérimentations menées par nos partenaires du LP3C/Loustic, il a été constaté que cette zone (qui intégrait aussi un début de feedback de guidage), pouvait comporter trop d'information, de fait nous l'avons simplifiée en discussion avec nos partenaires pour aboutir à la jauge. Bien entendu, des expérimentations sont envisageables à moyen terme pour déterminer la meilleure forme à cet espace.

Une fois que l'élève corrige ses erreurs, la jauge de complétion se met à jour, et la couleur de l'élément passe au vert, si il est résolu (*cf.* figure 11.11). Ce retour est donc purement descriptif, au sens où il alerte l'élève sur d'éventuelles erreurs, mais ne prodigue pas de conseil sur la procédure à suivre pour la correction des erreurs.

Intéressons nous maintenant à la production de conseils qui est réalisée *via* des feedbacks de guidage, que nous décrivons dans la sous-section suivante.

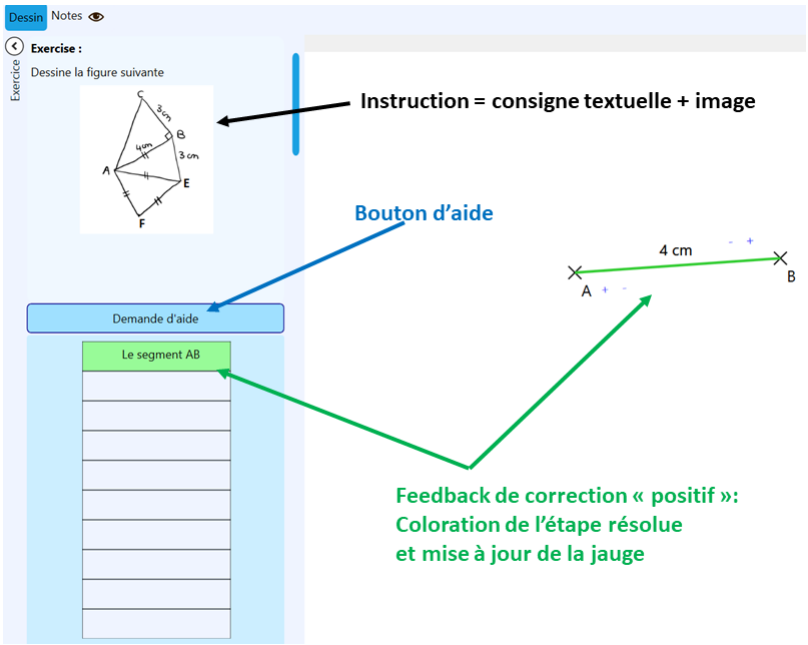


FIGURE 11.11 – Feedback "positif" : validation par le tuteur de l'étape de résolution de l'apprenant

11.3.3 Feedback de guidage

Le feedback de guidage peut avoir plusieurs rôles :

— Donner une piste sur la prochaine étape à résoudre ;
— Informer sur l'action de dessin à réaliser pour corriger les erreurs ;
— Indiquer le fait que l'élève est dans une impasse et donc qu'un retour en arrière est nécessaire.

La figure 11.12 illustre le premier rôle du feedback du guidage. Après avoir complété les quatre premières étapes de l'exercice, l'élève demande explicitement de l'aide au tuteur en appuyant sur le bouton *demande d'aide*. Le module expert analyse l'état de résolution représenté par le graphe de connaissance, et synthétise un plan de résolution complet à partir de la production de l'élève. Il extrait de ce plan la prochaine action à effectuer par l'élève pour la lui indiquer et ainsi lui permettre d'avancer dans le processus de résolution de l'exercice. Dans l'exemple, la bulle d'information apparaît sur la zone de dessins pour lui indiquer qu'il doit utiliser le compas pour dessiner le triangle isocèle ABE.

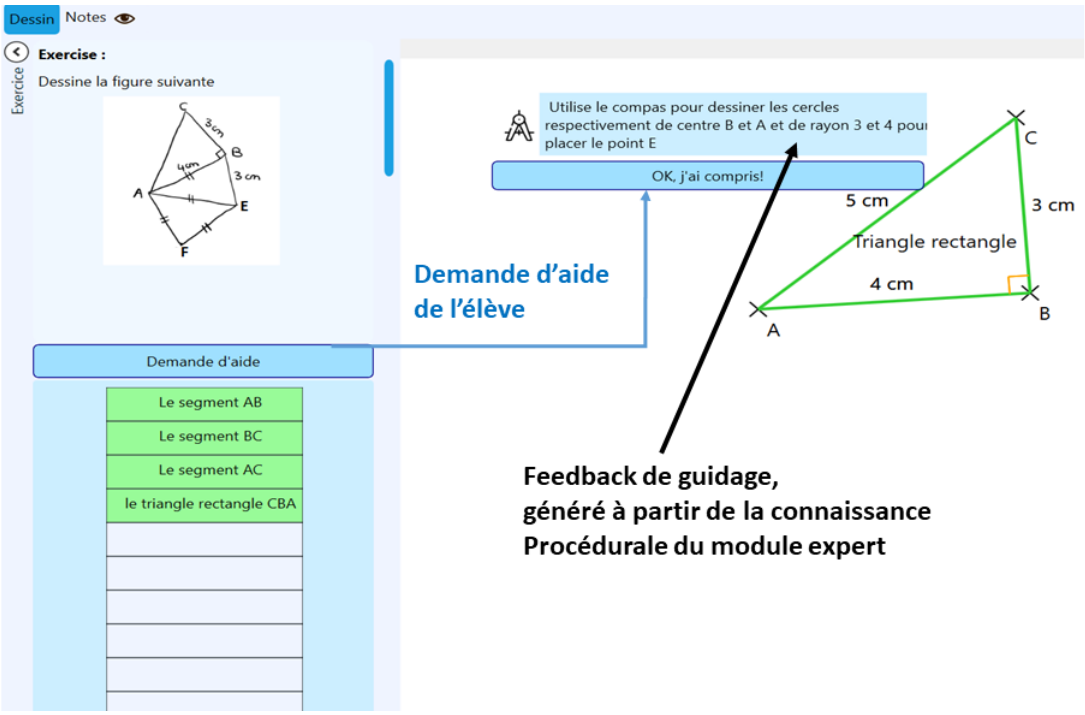


FIGURE 11.12 – Feedback de guidage : indication sur la prochaine étape

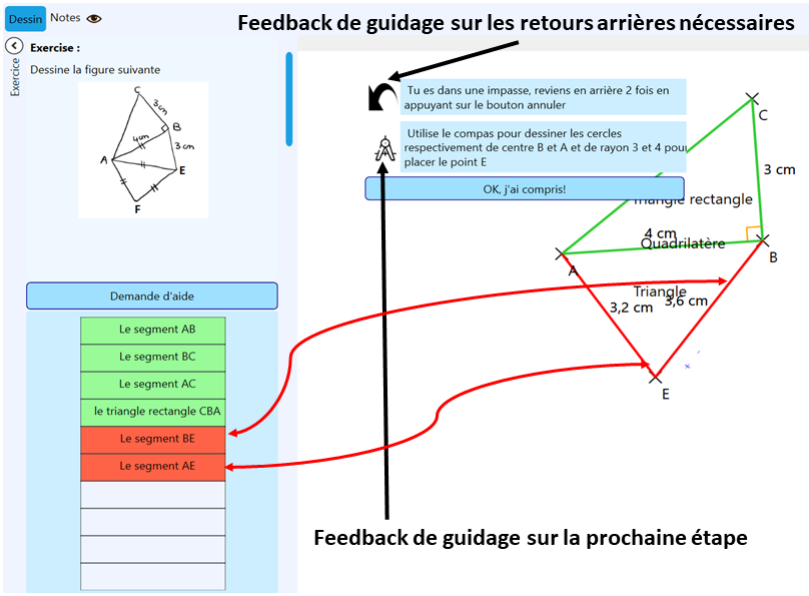


FIGURE 11.13 – Feedback de guidage : indication sur les actions de retours en arrière et sur la prochaine étape

La figure 11.13 illustre les deux autres rôles mentionnés plus haut. Dans cet exemple, l'élève est dans une situation de blocage. Les couleurs des segments [BE] et [AE] ainsi que la jauge de complétion indiquent la présence d'erreurs sur les longueurs. A la demande de l'élève, le module expert analyse son état de résolution et essaye de trouver un plan solution adapté. L'élève est dans une impasse, *i.e.* un état non soluble, puisqu'il n'est pas possible de corriger une figure fermée (ABE). Le module expert effectue un premier retour en arrière, puis un second (il n'est pas possible de résoudre le noeud BE sans utiliser le compas). Revenu sur un état soluble, le module expert peut alors élaborer un plan solution. Les actions de retour en arrière ainsi que la première action de dessin du plan solution sont ensuite affichées sous forme textuelle à l'élève.

Après avoir présenté les typologies de feedbacks générés par le système tutoriel à l'élève, nous nous intéressons maintenant à la valeur ajoutée qu'IntuiGeo peut apporter à l'enseignant en termes d'analyse fine de la production de l'élève, notamment la capacité de revoir la stratégie de résolution qu'il a suivie.

11.4 Possibilités d'évaluation pour l'enseignant

Pour permettre à l'enseignant d'évaluer par lui même le résolution de l'élève, il a la possibilité de revoir intégralement la procédure suivie par celui-ci *a posteriori*, en rejouant étape par étape le processus de résolution sur l'interface à l'aide d'un bouton replay/rejouer. Ceci lui permettra de voir exactement les outils utilisés, les typologies d'erreurs, et les stratégies choisies par l'élève.

Au delà de permettre à l'enseignant de rejouer dynamiquement la réalisation de l'élève, nous avons aussi développé la génération d'un rapport (PDF) de réalisation de l'exercice. Ce rapport résume les réalisations de l'élève, comme illustré dans la figure 11.14 pour la résolution d'un exercice de construction de parallélogramme.
Ce rapport contient :

- La figure finale réalisée par l'élève, ce qui permet de savoir instantanément si l'exercice a été résolu ;
- L'évaluation par le tuteur de la production finale dans la section *erreurs finales* ;
- des feedbacks de guidage générés chaque fois que l'élève demande de l'aide au système dans la section *détails de l'aide fournie.*

Nous pouvons voir dans cet exemple que l'aide a été demandée deux fois : chaque feedback de guidage est associé à un état de résolution, représentée par la figure en dessous du feedback textuel. Cela permet à l'enseignant de voir comment a réagit l'élève à l'aide qui lui a été fournie.

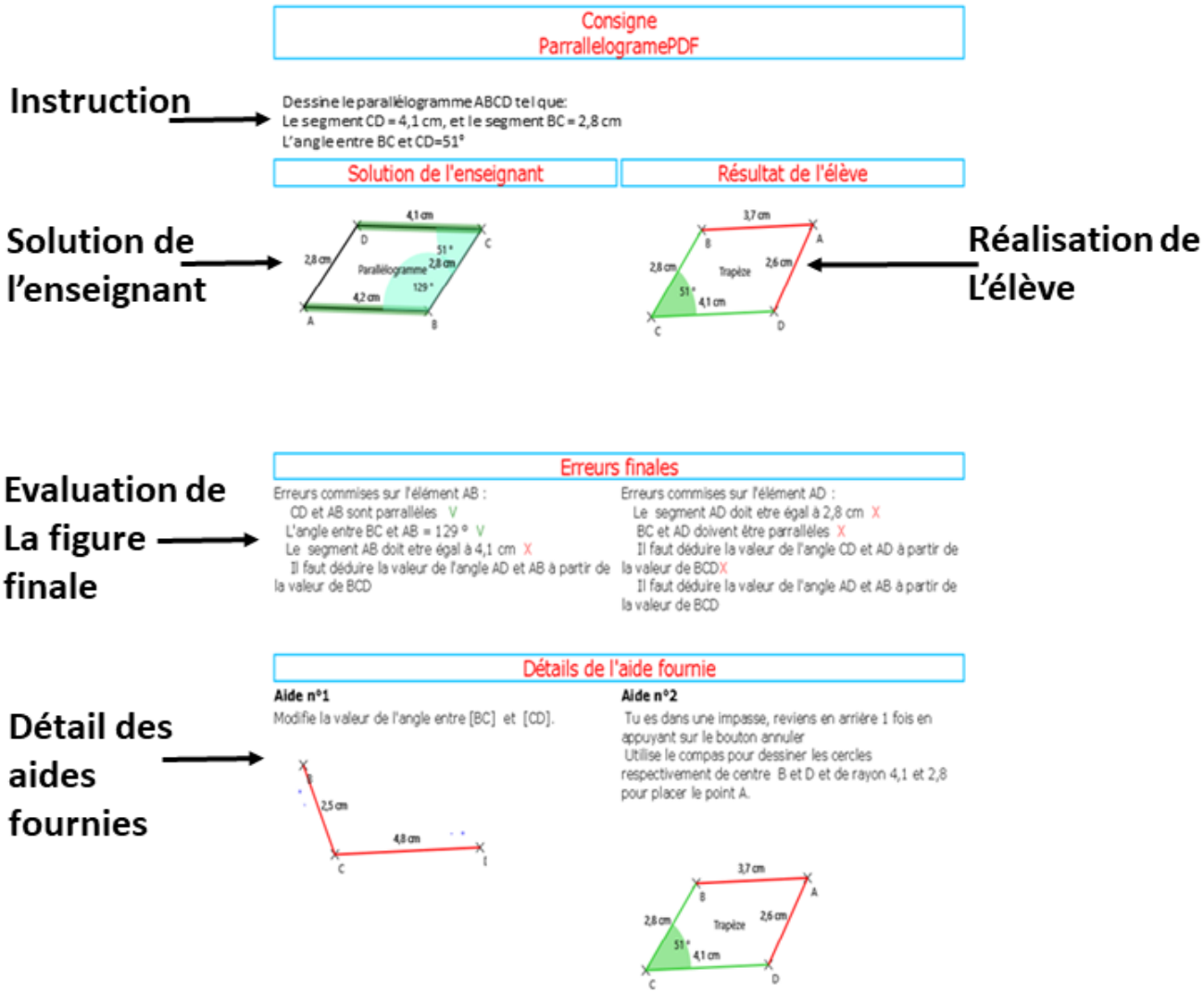


FIGURE 11.14 – Rapport généré par le système, résumant la réalisation d'un exercice de géométrie

11.5 Bilan

Dans ce chapitre, nous avons présenté l'ensemble des fonctionnalités et l'ergonomie de notre prototype IntuiGeo. Nous nous sommes intéressés à : l'interaction utilisateur avec les fonctions d'édition et la modélisation des outils virtuels, la typologie des feedbacks générés, et l'analyse a posteriori de la production de l'élève.

Dans le chapitre 12 suivant, nous présentons l'impact pédagogique d'IntuiGeo sur les performances des élèves, notamment en termes de transférabilité de l'apprentissage entre support numérique et papier.

Chapitre 12

Impact pédagogique d'IntuiGeo

Notre collaboration avec le LP3C/Loustic de l'université de Rennes 2, et l'apport des enseignants volontaires de l'académie de Rennes (en Bretagne) nous ont permis de mener une étude d'impact pour évaluer l'apport de notre tuteur sur la performance des élèves en termes d'apprentissage. Cette étude vient couronner un long processus de mise au point et d'optimisations ergonomiques, dont nous avons parlé dans le chapitre précédent, qui a contribué à l'évolution du système.

Les expérimentations en classe (conduites par les experts en psychologie expérimentale et en science des usages du LP3C/Loustic) ont un triple objectif :

1. Pouvoir comparer la performance de l'élève relativement au support : traditionnel papier-crayon ou numérique tablette stylet ;
2. Étudier l'effet du feedback de correction sur le taux de réussite de l'élève ;
3. Évaluer la transférabilité de l'apprentissage du support numérique vers le support papier.

Lors de l'expérimentation réalisée en Décembre 2019, seul l'impact du feedback correctif a été évalué. L'expérimentation pour l'évaluation du feedback de guidage, prévue en mars 2020 n'a pu être réalisée étant donnée la crise sanitaire.
Les hypothèses de départ, *i.e.* avant les expérimentations, sont les suivantes :

1. Nous pensons que le feedback généré par le tuteur entraînera une meilleure performance des élèves par rapport à la condition traditionnelle ;
2. Nous estimons que cette meilleure performance s'étendrait aussi au transfert de l'apprentissage du numérique vers le papier, vu que notre système simule le support traditionnel ;
3. D'un point de vue subjectif, nous estimons que l'utilisation de la tablette stimulerait le sentiment d'intérêt pour la géométrie des élèves, et que par contre, la difficulté perçue à la résolution des exercices serait plus grande.

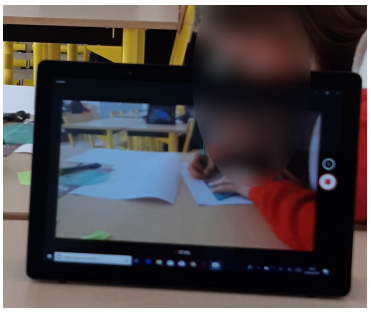
12.1 Protocole de l'étude

Les participants à l'expérimentation de décembre 2019 sont des élèves en cinquième dans le collège La Roche aux Fées (35) [1], de l'académie de Rennes. Un total de 85 élèves a participé à l'étude (39 filles et 45 garçons). Ils ont été affectés, de façon aléatoire, à trois groupes :

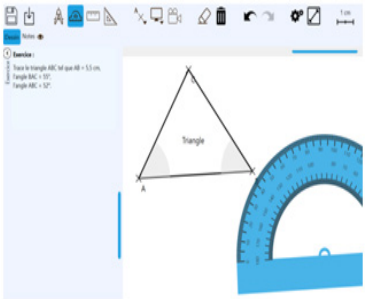
1. Un groupe de contrôle : réalisation de figures sur support papier (condition traditionnelle), composé de 27 élèves (*cf.* figure 12.1.a) ;
2. Un groupe *IntuiGeo sans feedback* : utilisation d'IntuiGeo comme éditeur de figures, composé de 28 élèves (*cf.* figure 12.1.b) ;
3. Un groupe *IntuiGeo avec feedback* : génération par le tuteur de feedback correctif en temps-réel, composé de 30 élèves (*cf.* figure 12.1.c).

Les trois conditions sont illustrées dans la figure 12.1. Pour le groupe *tablette avec feedback*, les segments se colorient automatiquement en vert si la longueur est correcte, en rouge sinon, et de même pour les angles (*cf.* figure 12.1.c).

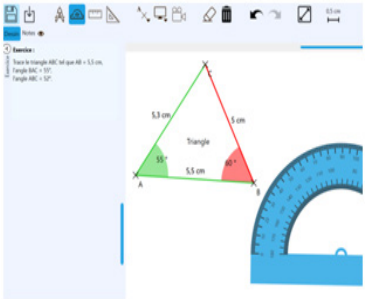
L'intérêt du groupe *tablette sans feedback* est d'**isoler l'effet du feedback de l'effet du support utilisé**. De plus, les informations visuelles (longueur de segment et valeur d'angle) ne sont présentes que sur les derniers éléments tracés par l'élève. C'est la condition qui s'approche le plus de la condition traditionnelle, on pourra donc voir s'il y a un effet intrinsèque au support (la tablette) sur la performance des élèves.



(a) Groupe contrôle : papier-crayon



(b) Groupe tablette sans feedback



(c) Groupe tablette avec feedback

FIGURE 12.1 – Répartition des élèves selon trois groupes/conditions pour l'étude

1. Nous remercions encore une fois les enseignants et les élèves de leur participation

12.2 Procédure d'évaluation des performances

Pour mener à bien cette étude, il est important de respecter une méthodologie scientifique solide. En ce sens, l'expertise du LP3C/loustic a été primordiale. Nous suivons une procédure d'évaluation en trois phases (chacune est détaillée dans une sous-section ci-après) :

1. Phase d'évaluation du niveau initial ;
2. Phase d'entraînement ;
3. Phase d'apprentissage ;
4. Phase de transfert.

12.2.1 Évaluation du niveau initial

Dans la première étape, nous avons évalué le niveau des participants en géométrie. L'objectif de cette phase est de s'assurer qu'il n'y a pas de différence significative préalable quant au niveau en géométrie entre les trois groupes.

Pour cela, nous avons d'une part demandé à chaque élève de se positionner sur une échelle de Lickert [MJ71] de 1 à 7 (7 voulant dire *totalement d'accord* et 1 voulant dire *pas du tout d'accord*) par rapport à l'affirmation : *je me considère bon élève en géométrie.*

D'autre part, tous les élèves ont eu quatre minutes pour résoudre un exercice de construction de figure géométrique dans la condition traditionnelle papier-crayon, avec l'utilisation des outils classiques (compas, règle, etc). Le problème en question était de construire un triangle rectangle, la consigne était la suivante : *Trace un triangle rectangle ABC en B tel que BC = 6cm et l'angle BAC = 40 degrés.*

12.2.2 Phase d'entraînement

Pour la phase d'entraînement, tous les élèves (de tous les groupes) effectuent des tâches de construction géométrique sur leur support dédié (dessin de segments, utilisation des outils virtuels/classiques). Cette phase est importante, surtout pour les deux groupes utilisant la tablette stylet, cela permet de s'assurer de leur compréhension de l'utilisation du logiciel. Les élèves ont eu six minutes d'entraînement pour :

— Dessiner un segment à main levée ;
— Dessiner un segment en utilisant la règle ;
— Dessiner un angle en utilisant le rapporteur ;

— Supprimer leur dessin en utilisant la gomme.

Les élèves du groupe contrôle ont effectué les mêmes tâches sur support papier.

12.2.3 Phase d'apprentissage

Cette phase permet d'évaluer les performances des élèves des trois groupes. L'objectif ici est de travailler sur trois exercices de construction de triangles scalènes :

— Exercice 1 : trace le triangle ABC tel que AB = 5,5 cm, BAC = 55°, ABC = 52° ;
— Exercice 2 : trace le triangle ABC tel que AB = 6,8 cm, BAC = 31°, ABC = 40° ;
— Exercice 3 : trace le triangle ABC tel que AB = 7 cm, BAC = 43°, ABC = 50°.

Chaque élève a six minutes pour réaliser un exercice. Notons que les exercices ont la même difficulté.

12.2.4 Phase de transfert

La dernière étape est la mesure d'impact sur l'apprentissage *via* la phase de transfert, où tous les groupes effectuent les exercices sur le support papier. Cette dernière étape va permettre l'analyse de la transferabilité de l'apprentissage entre support numérique et support papier. Les élèves ont eu deux exercices à effectuer dans cette phase :

— une tâche de transfert dite "proche" avec l'exercice suivant : Trace le triangle ABC tel que AB = 8 cm, BAC = 70°, ABC = 32° ;
— une tâche de transfert dite "lointaine" avec l'exercice suivant : Trace le triangle rectangle ABC en B tel que AB = 4cm et l'angle BAC = 50°.

La première tâche est dite proche parce que l'exercice est similaire à ceux proposés dans la phase d'apprentissage, tandis que la seconde implique des connaissances qui n'ont pas été mises en oeuvre durant la phase d'apprentissage.

12.3 Résultats

Pour l'évaluation des performances lors des trois phases (évaluation du niveau initial, phase d'apprentissage, et phase de transfert), un codage binaire a été considéré (0 si l'élève n'a pas réussi la figure, 1 sinon).

12.3.1 Résultats sur l'évaluation du niveau initial

Sur les 82 élèves ayant répondu à la question *j'ai un bon niveau en géométrie*, il n'y a pas de différence significative entre les trois groupes.

— Pour le groupe papier-crayon : M (Moyenne) = 4,19 tandis que ET (Écart Type) = 1.21 ;
— Pour le groupe IntuiGeo sans feedback : M = 4,15, ET = 1,70 ;
— Pour le groupe IntuiGeo avec feedback : M = 4,61, ET = 1,6.

Sur la réalisation du triangle rectangle, il n'y a pas de différence significative non plus entre les trois groupes. Notons cependant un pourcentage de succès très bas, respectivement de 14.81%, 3.57%, et 13,33% pour les trois groupes. Il est intéressant de remarquer que des élèves de cinquième affichent des difficultés avec une figure qui peut pourtant paraître simple.

Nous pouvons conclure de cette phase, à l'aube de l'évaluation des performances sur les différents supports, que le niveau des trois groupes est homogène.

12.3.2 Résultats sur la phase d'apprentissage

Le tableau 12.1 présente le pourcentage de succès des élèves des trois pour chacun des trois exercices de construction de triangle. On remarque que la performance est similaire pour les élèves des deux groupes "papier-crayon" et "IntuiGeo sans feedback". De ce résultat, nous pouvons conclure que le support en lui même n'a pas d'impact significatif sur la performance des élèves. Nous remarquons par ailleurs que la performance des participants appartenant aux troisième groupe ("IntuiGeo avec feedback") est bien meilleure, pour chaque exercice, que celle des autres groupes. Cela nous permet de conclure que la **génération de feedback correctif en temps-réel a un impact significatif sur la performance des élèves**.

Table 12.1 – Performance des trois groupes lors de la phase d'apprentissage

Exercice	Groupe papier-crayon	Groupe IntuiGeo sans feedback	Groupe IntuiGeo avec feedback
exercice 1	23.08 %	17.86%	**55.17 %**
exercice 2	23.08 %	21.43%	**86.21 %**
exercice 2	23.08 %	22.22%	**92.86 %**

Un autre point intéressant de ces résultats est l'évolution des performances au fur et et à mesure de la réalisation des exercices telle qu'illustrée sur la figure 12.2. Nous pouvons remarquer que le pourcentage de réussite évolue rapidement pour le groupe tablette avec

feedback pour atteindre un niveau de 92.86 % pour le troisième exercice. Par contraste, nous constatons une stagnation, ou au plus une très légère progression, pour les deux autres groupes, avec un taux de réussite pour le troisième exercice qui tourne autour des 20 %.

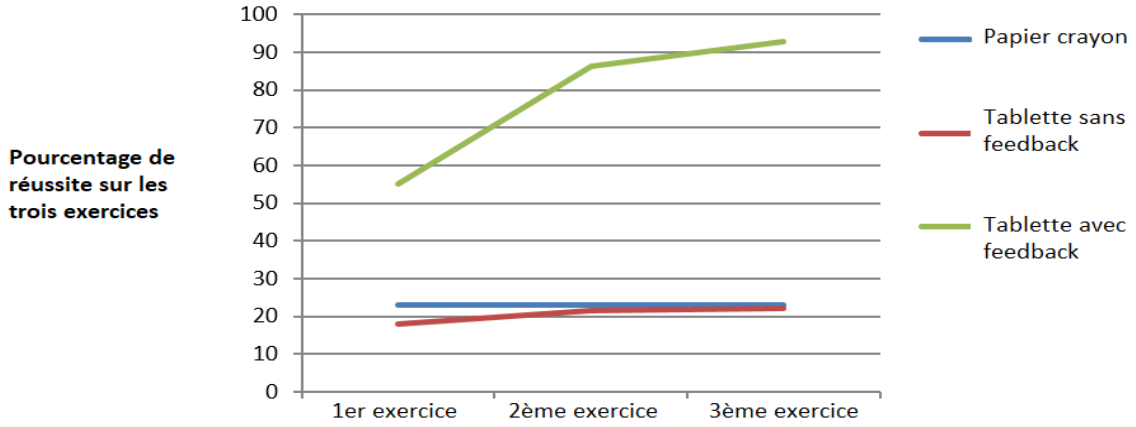


FIGURE 12.2 – Évolution de la performance au fur et et à mesure de la réalisation des exercices

Les élèves progressent donc bien dans leur apprentissage grâce au mode "IntuiGeo avec feedback", les feedbacks immédiats et personnalisés s'avèrent très efficaces.

Maintenant, il reste à démontrer que ce gain en terme d'apprentissage dans la modalité "IntuiGeo avec feedback" sera bien transféré lorsque les élèves reviennent vers la modalité traditionnelle papier-crayon.

12.3.3 Résultats sur la phase de transfert

Comme indiqué plus tôt, pour la phase de transfert, les élèves des trois groupes ont réalisé deux exercices sur *support papier*. Le tableau 12.2 présente les résultats de performance des élèves pour cette phase.

TABLE 12.2 – Performance sur les tâches de transfert

Tâche	Groupe papier-crayon	Groupe tablette sans feedback	Groupe avec feedback
Triangle scalène	25.93%	25%	60%
Triangle rectangle	22.22%	10.71%	23.33%

Pour la tâche de transfert proche (triangle scalène), nous remarquons une différence significative en terme de performance entre le groupe tablette avec feedback et les deux autres. En effet, le pourcentage de succès des élèves de ce groupe se situe à 60 %, contre 26 et 25% pour les deux autres. Ceci est un résultat très important, dans la mesure où les compétences acquises sur format numérique à l'aide de la correction du tuteur **sont transférées sur le support papier**.

Pour la tâche de transfert lointaine toutefois, nous ne remarquons pas de différence significative entre les trois groupes. Ce résultat est logique puisque les compétences requises pour réussir cette tâche n'ont pas été mises en oeuvre durant la phase d'apprentissage.

12.3.4 Mesures subjectives

En plus de ces évaluations factuelles, à la fin de l'expérimentation, les élèves ont répondu à un questionnaire pour faire connaître leur sentiment sur la difficulté perçue, et leur intérêt pour la tâche, selon le support attribué. Comme pour le questionnaire de l'évaluation du niveau initial, une échelle de Lickert de 1 à 7 a été utilisée pour ces mesures. Le tableau 12.3 présente ces mesures subjectives.

Table 12.3 – Difficulté perçue et intérêt pour la géométrie après réalisation des exercices

Mesure	Groupe papier-crayon	Groupe tablette sans feedback	Groupe avec feedback
Difficulté perçue	M = 3.68 ET = 1.87	M = 2.75 ET = 1.76	**M = 2.39 ET = 1.52**
Intérêt	M = 4.92 ET = 1.24	M = 5.92 ET = 1.21	**M = 6.08 ET = 1.22**

Nous pouvons voir que c'est toujours le groupe ayant utilisé IntuiGeo avec feedback qui a les meilleurs scores. Le fait que la tablette attise l'intérêt des élèves pour la géométrie était attendu, et les deux groupes utilisant ce support ont des indicateurs très proches (5.92 et 6.08). Cependant, le fait que les élèves du groupe feedback aient moins ressenti la difficulté de la tâche que le groupe papier-crayon (M = 2.39 versus M = 3.68) parait quelque peu surprenant. Une explication plausible pourrait résider dans le fait que nous simulons, dans notre tuteur orienté stylet, l'approche traditionnelle papier-crayon, avec des outils virtuels réalistes. Cela rend le processus de familiarisation avec l'outil assez rapide.

12.4 Bilan

Nous avons présenté dans ce chapitre notre étude en écosystème portant sur l'impact pédagogique d'IntuiGeo. Les résultats ont démontré que le pourcentage de succès pour le groupe tablette avec feedback était significativement meilleur que les deux autres groupes. Nous n'avons pas mesuré de différence significative entre le groupe de contrôle (papier-crayon) et le groupe tablette sans feedback. Il est donc clair que c'est le feedback correctif généré en temps-réel qui fournit une plus-value pédagogique à l'élève, et non pas le support.

Un autre résultat intéressant est la progression des élèves du groupe "IntuiGeo avec feedback" au cours de la réalisation des trois exercices. Cette progression montre une compréhension des retours du tuteur et une acquisition des compétences requises pour résoudre des problèmes de même type. Un point qui nous parait très intéressant est le fait que cette progression, induite par le feedback, se transfère lorsque l'élève change de support et revient à la condition traditionnelle papier-crayon. Cela démontre que notre logiciel IntuiGeo peut être utilisé comme complément de cours de géométrie. Ces résultats ont été publiés dans [4] et soumis pour une revue internationale [7].

Il aurait été intéressant d'évaluer l'impact pédagogique du feedback de guidage généré par le module expert, en particulier par rapport à la connaissance procédurale qu'il représente et qui pourrait amener les élèves à acquérir de nouvelles compétences (comment et quand utiliser un compas virtuel par exemple). Comme évoqué précédemment, l'évaluation pédagogique du guidage n'a pas pu être réalisé en raison de la crise sanitaire mais fait partie des perspectives de ces travaux.

Il aurait été tout aussi intéressant de comparer notre outil à d'autres logiciels de géométrie dynamique. À notre connaissance cependant, aucun de ces logiciels ne fournit de feedback correctif en temps-réel.

Bien entendu, il est légitime de se demander si ces résultats positifs tiennent sur la durée. Pour cela il faudrait concevoir une expérimentation sur plusieurs mois, expérimentation qui ne pouvait être envisagée sur la durée de la thèse entre autres en raison des contraintes liées à la crise sanitaire. Cela pourra faire l'objet de travaux futurs.

Conclusion et perspectives

Conclusion

Ces travaux de thèse ont porté sur la conception d'un système tutoriel intelligent orienté stylet, pour l'apprentissage de la géométrie au collège. L'objectif de départ était triple :

1. Simuler l'approche traditionnelle papier crayon en profitant de l'interaction stylet-doigts offerte par la tablette, pour offrir à l'élève une expérience intuitive et rendre le support numérique transparent à l'apprenant ;
2. Garantir la transférabilité de l'apprentissage du support numérique vers le support papier ;
3. Stimuler l'apprentissage actif et améliorer les performances des élèves, en générant des feedbacks personnalisés de correction en temps-réel ainsi que des feedbacks de guidage à la demande.

Nous avons conçu IntuiGéo (Intuitive Geometry), une approche originale se basant sur une combinaison d'approches de reconnaissance de formes et d'approches provenant du domaine des STI (Systèmes Tutoriels Intelligents). En prenant en compte ces objectifs de départ, les contributions de cette thèse se présentent sous deux axes.

Le premier axe est l'extension du formalisme grammatical GMC-PC en grammaire hiérarchique GMC-HPC pour faire face à la complexité structurelle des documents traités. Cette extension a permis l'adaptation du formalisme à un contexte d'e-apprentissage où la contrainte d'interprétation à la volée des tracés de l'élève est primordiale. La définition de niveaux d'interprétation a permis d'introduire la notion de couches sémantiques (présentes dans le domaine de la géométrie, mais aussi dans d'autres domaines tel que l'architecture) tout en contrôlant la complexité de la combinatoire du processus d'analyse. Nous avons souhaité mettre en place des stratégies permettant de conserver l'aspect générique de la grammaire. La généricité de cette contribution a été démontrée sur un autre domaine d'application, la composition de plans d'architecture, où la sémantique est représentée par la capacité à reconnaître des pièces. Le moteur adapté à la géométrie a fait l'objet

d'un transfert industriel.

Nous avons aussi étendu l'analyseur associé au formalisme, en introduisant la notion d'analyseur sémantique contextuel, capable de détecter les patterns particuliers (les cycles) dans le document, sans pour autant causer une explosion combinatoire, ce qui était le cas dans le formalisme de base. Cette extension a été la base de notre moteur de reconnaissance 2D, la première brique de notre tuteur orienté stylet.

Le deuxième axe de cette thèse a consisté à proposer une approche originale pour la brique tutorielle de notre système. L'étude de la littérature nous a permis de dégager les grandes familles d'approches, sans qu'aucune d'elles ne soient directement adaptées à notre besoin. Notre moteur de supervision, responsable de la partie tutorielle d'IntuiGeo, est basé sur une méthode originale combinant les caractéristiques des tuteurs à base de contraintes et des tuteurs à base de règles. Nous proposons un mode auteur intuitif où il suffit à l'enseignant de dessiner un exemple solution pour générer un modèle complet du problème. Ce modèle est représenté par un graphe de connaissance, contenant tous les éléments géométriques et les contraintes structurelles et géométriques associées. L'évaluation à la volée et en temps-réel des actions de l'apprenant, éléments géométriques interprétés par le moteur de reconnaissance 2D, est gérée par le module de l'apprenant et s'appuie sur la mise en correspondance, à la volée, de ces éléments avec le graphe de connaissance. Cela permet de mettre à jour l'état de résolution de l'élève et de générer un feedback de correction si le tuteur détecte une erreur.

Le module expert, quant à lui, est capable de synthétiser des stratégies de résolution à partir de n'importe quel état de résolution de l'apprenant afin de lui fournir des feedbacks de guidage, qui peuvent être des indications sur les prochaines étapes, des conseils de retour en arrière en cas d'impasse, ou plus simplement des pistes sur comment corriger ses erreurs. Nous avons défini pour ce faire un environnement de planification dynamique, avec des actions de dessin qui étendent la définition des actions de planification classiques en y incluant des tracés artificiels que le module réalise sur le document pour simuler le comportement de l'expert.

La collaboration avec le laboratoire LP3C et la plate-forme Loustic de l'université Rennes 2 au sein du projet ACTIF a permis de suivre un processus de conception centré-utilisateur, avec la participation d'élèves et de professeurs volontaires de l'académie de Rennes. L'IHM et l'ergonomie d'IntuiGeo ont donc évolué itérativement au fil des expérimentations grâce aux retours des différents participants. La dernière version du tuteur a fait l'objet d'une étude d'impact en écosystème (en classe). L'objectif était d'évaluer

l'impact pédagogique de l'outil, en nous basant sur les performances des élèves l'ayant utilisé sur des exercices de construction par rapport à un groupe de contrôle d'élèves qui ont travaillé à partir d'une approche classique papier/crayon. Les résultats ont démontré que l'utilisation d'IntuiGeo a un impact positif, et que le feedback de correction personnalisé délivré en temps-réel permet aux élèves de se rendre compte de leurs erreurs et de les corriger. Cette étude a aussi démontré la transférabilité de l'apprentissage, puisque les élèves ayant utilisé l'outil ont aussi une meilleure performance lorsqu'ils reviennent sur le support papier.

Cette étude d'impact, réalisée à l'école, démontre le bien fondé du tuteur intelligent IntuiGeo et l'importance des objectifs que nous nous étions fixés au départ de cette thèse : la simulation de l'approche traditionelle papier/crayon, la garantie de la transférabilité de l'apprentissage, et la génération de feedbacks personnalisés immédiats pour encourager l'apprentissage actif. Pour cela, nous avons conçu, étendu et combiné plusieurs approches (reconnaissance à la volée de tracés manuscrits, modélisation de la connaissance et de l'apprenant, planification), en faisant face à la combinatoire liée à la complexité de tache pour garantir des réponses en temps-réel.

Nous avons aujourd'hui un logiciel d'aide à la l'apprentissage de la géométrie opérationnel grâce à la conception centrée utilisateur épaulée par le LP3C/Loustic. Et au delà de ce domaine applicatif, le formalisme GMC-PC pour l'interprétation à la volée de documents manuscrits a été étendu et consolidé, ce qui ouvre son adaptation à d'autres domaines structurés complexes.

Perspectives

Ces travaux peuvent mener à plusieurs perspectives. Une première est liée à l'enrichissement du mode auteur. Pour l'instant, l'exercice est généré à partir de la figure solution de l'enseignant qui peut ensuite modifier la consigne textuelle générée. Aujourd'hui, nous travaillons à lui donner plus de contrôle sur le paramétrage de l'exercice. En effet, un enseignant pourrait vouloir inviter l'élève à utiliser un rapporteur au lieu d'un compas pour construire un triangle équilatéral par exemple. Cela implique d'interdire au module expert de proposer les stratégies simulant l'utilisation du compas pour la résolution du problème. Cela implique aussi une modification de l'évaluation à la volée des actions de l'élève par le module apprenant. En effet, l'évaluation consiste actuellement à vérifier que les contraintes du problèmes sont satisfaites, avec cette extension l'évaluation devra inclure une vérifi-

cation aussi centrée sur le processus de réalisation de l'action. Une deuxième perspective est liée à l'enrichissement du module de l'apprenant. Pour l'instant, ce module possède une connaissance court-terme de l'élève, *i.e.* son état de résolution d'un exercice donné. Nous allons faire évoluer ce module pour inclure une connaissance long-terme, le tuteur aura donc un historique des exercices de l'élève et nous souhaitons pouvoir construire un modèle plus complet de l'élève, qui inclut ses compétences et ses faiblesses. L'objectif de cette connaissance à long-terme est de pouvoir proposer automatiquement des exercices adaptés aux niveau de l'apprenant, la répétition des problèmes en cas de compétences insuffisantes, etc...

Une troisième perspective serait de généraliser ce concept de tutorat orienté stylet pour d'autres domaines d'applications similaires à la géométrie de construction, que ce soit pour l'e-éducation où même pour des professionnels (comme pour les plans d'architecture par exemple). Le moteur de reconnaissance étant déjà générique, cette perspective implique de généraliser l'environnement de planification dynamique que nous avons proposé. Cela est tout à fait envisageable puisqu'à la manière de la formalisation de la connaissance dans le formalisme GMC-HPC, l'expert pourra définir les actions de dessin du module, en y associant des gestes qu'il tracera sur l'interface.

Enfin, une perspective à court terme est l'évaluation de l'impact des feedbacks de guidage sur la performance des élèves en suivant le protocole similaire à celui que nous avions établi pour évaluer l'impact des feedbacks de correction. Comme indiqué dans la section 12.4, il serait aussi intéressant de voir si les résultats positifs que nous avons constatés se confirment sur la durée en établissant des expérimentations qui durent plusieurs semaines, voire plusieurs mois si possible.

Publications de l'auteur

Conférences internationales

[1] Omar Krichen, Eric Anquetil, Nathalie Girard. IntuiGeo : Interactive tutor for online geometry problems resolution on pen-based tablets. European Conference on Artificial Intelligence (ECAI) 2020, Aug 2020, Santiago de Compostela, Spain, pp.1842 - 1849. https://hal.archives-ouvertes.fr/hal-02544384v1.

[2] Omar Krichen, Nathalie Girard, Eric Anquetil. Extension of a bi-dimensional grammar for online interpretation of structured documents : application on architecture plans and geometry domains, ICFHR 2020, September 2020, Dortmund, Germany, pp.325 – 330, https://hal.archives-ouvertes.fr/hal-02929753.

[3] Omar Krichen, Nathalie Girard, Eric Anquetil, Simon Corbillé, Mickaël Renault. Real-time interpretation of hand- drawn sketches with extended hierarchical bi-dimensional grammar. The 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), Aug 2018, NIAGARA FALLS, United States. pp.273-278. https://hal.archives-ouvertes.fr/hal-01917888v1.

[4] Tiphaine Colliot, Omar Krichen, Nathalie Girard, Eric Anquetil, Eric Jamet. Learning with a geometry app on tablet : the critical role of real-time and corrective feedback. In Proceedings of EARLI SIG 2 conference (European Association for Research on Learning and Instruction), August 2020, Czech Republic.

[5] Omar Krichen, Nathalie Girard, Eric Anquetil, Mickaël Renault. Real-time interpretation of geometric shapes for digital learning. in Proc. ICPRAI (Int. Conf. on PR & AI), May 2018, Montreal, Canada. pp.31-36. https://hal.archives-ouvertes.fr/hal-01816617v1.

Revues internationales

Accepté

[6] Omar Krichen, Eric Anquetil, Nathalie Girard, Mickaël Renault. Online analysis of hand-drawn strokes for Geometry learning. Marleah Blom ; Nicola Nobile ; Ching Y Suen. Frontiers in Pattern Recognition and Artificial Intelligence, 5, World Scientific, pp.129-149, 2019, Language Processing, Pattern Recognition, and Intelligent Systems, 9789811203527 `https://hal.archives-ouvertes.fr/hal-02162405v1`.

Soumis, en cours de relecture

[7] Tiphaine Colliot, Omar Krichen, Nathalie Girard, Eric Anquetil, Eric Jamet. Learning with a Geometry App on a Tablet : Critical Role of Realtime and Corrective Feedback. **Submitted to** : Educational Technology Research and Development, 2020.

Conférences nationales

[8] Simon Corbillé, Eric Anquetil, Omar Krichen, Nathalie Girard, Mickaël Renault. IntuiGeo, éditeur de figures géométriques à main levée pour l'apprentissage de la géométrie sur tablette. 30 ème conférence francophone sur l'interaction homme machine, Octobre 2018, Brest, France, 2 p. `https://hal.archives-ouvertes.fr/hal-01900050`.

Symposiums et colloques nationaux (Présentations/Posters)

[9] Omar Krichen, Nathalie Girard, Eric Anquetil, Simon Corbillé. IntuiGéo : tuteur intelligent pour l'apprentissage de la géométrie sur tablette. Second colloque scientifique e-FRAN, Oct 2019, Paris, France. `https://hal.archives-ouvertes.fr/hal-02500363v1`

[10] Omar Krichen, Eric Anquetil, Nathalie Girard. Interprétation temps-réel de la production de schémas géométriques pour le Digital Learning. premier colloque scientifique e-FRAN, Jan 2018, Paris, France. `https://hal.archives-ouvertes.fr/hal-01819549v1`

[11] Omar Krichen, Eric Anquetil, Nathalie Girard, Simon Corbillé. IntuiGéo : tuteur intéractif pour la résolution de problèmes de géométrie sur tablette. SIFED (Symposium International Francophone sur l'Écrit et le Document), Jun 2019, Nancy, France. `https://hal.archives-ouvertes.fr/hal-02978726`

Bibliographie

[Ale+09] Vincent Aleven et al., « A New Paradigm for Intelligent Tutoring Systems : Example-Tracing Tutors. », in : *I. J. Artificial Intelligence in Education* 19 (jan. 2009), p. 105-154.

[Ale10] Vincent Aleven, « Rule-Based Cognitive Modeling for Intelligent Tutoring Systems », in : *Advances in Intelligent Tutoring Systems*, sous la dir. de Roger Nkambou, Jacqueline Bourdeau et Riichiro Mizoguchi, Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, p. 33-62, isbn : 978-3-642-14363-2, doi : 10.1007/978-3-642-14363-2_3, url : https://doi.org/10.1007/978-3-642-14363-2_3.

[Alv+14] Chris Alvin et al., « Synthesis of Geometry Proof Problems », in : *AAAI'14 Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, Association for the Advancement of Artificial Intelligence (AAAI), juill. 2014, p. 245-252, url : https://www.microsoft.com/en-us/research/publication/synthesis-geometry-proof-problems/.

[AMP04] Chadia Abras, Diane Maloney-krichmar et Jenny Preece, « User-Centered Design », in : *In Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks : Sage Publications*, Publications, 2004.

[AMS09] Vincent Aleven, Bruce Mclaren et Jonathan Sewall, « Scaling Up Programming by Demonstration for Intelligent Tutoring Systems Development : An Open-Access Web Site for Middle School Mathematics Learning », in : *TLT* 2 (avr. 2009), p. 64-78, doi : 10.1109/TLT.2009.22.

[And83] John R. Anderson, *The Architecture of Cognition*, Excerpts available on Google Books (see link below). For more information, go to publisher's website : http ://www.routledge.com/books/details/9780805822335/, Harvard University Press, 1983, 345 pages, url : https://hal.archives-ouvertes.fr/hal-00699788.

[AS10] F. Alvaro et J. A. Sanchez, « Comparing Several Techniques for Offline Recognition of Printed Mathematical Symbols », in : *2010 20th International Conference on Pattern Recognition*, 2010, p. 1953-1956.

[ÁSB14] Francisco Álvaro, Joan-Andreu Sánchez et José-Miguel Benedí, « Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models », in : *Pattern Recognition Letters* 35 (2014), Frontiers in Handwriting Processing, p. 58-67, issn : 0167-8655, doi : https://doi.org/10.1016/j.patrec.2012.09.023, url : http://www.sciencedirect.com/science/article/pii/S016786551200308X.

[ÁSB16] Francisco Álvaro, Joan-Andreu Sánchez et José-Miguel Benedí, « An integrated grammar-based approach for mathematical expression recognition », in : *Pattern Recognition* 51 (2016), p. 135-147, issn : 0031-3203, doi : https://doi.org/10.1016/j.patcog.2015.09.013, url : http://www.sciencedirect.com/science/article/pii/S0031320315003441.

[Ati+14] Olufunmilola Atilola et al., « Mechanix : A natural sketch interface tool for teaching truss analysis and free-body diagrams », in : *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 28.*2* (2014), p. 169-192, doi : 10.1017/S0890060414000079.

[Bal+03] Nicolas Balacheff et al., « Baghera Assessment Project, designing an hybrid and emergent educational society », in : (jan. 2003).

[BC15] Kaushal Kumar Bhagat et Chun-Yen Chang, « Incorporating GeoGebra into Geometry Learning-A lesson from India », in : *Eurasia Journal of Mathematics, Science and Technology Education* 11.*1* (2015), p. 77-86, issn : 1305-8215, doi : 10.12973/eurasia.2015.1307a, url : http://dx.doi.org/10.12973/eurasia.2015.1307a.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho et Yoshua Bengio, *Neural Machine Translation by Jointly Learning to Align and Translate*, 2014, arXiv : 1409.0473 [cs.CL].

[BD04] Kalina Bontcheva et Vania Dimitrova, « Examining the use of conceptual graphs in adaptative web-based systems that aid terminology learning », in : *International Journal on Artificial Intelligence Tools* 13.*02* (2004), p. 299-331,

DOI : 10.1142/S0218213004001569, eprint : https://doi.org/10.1142/S0218213004001569, URL : https://doi.org/10.1142/S0218213004001569.

[BJ13] Lars BOLLEN et Wouter van JOOLINGEN, « Simsketch : Multi-Agent Simulations Based on Learner-Created Sketches for Early Science Education », in : *IEEE Transactions on Learning Technologies* 6 (juill. 2013), p. 208-216, DOI : 10.1109/TLT.2013.9.

[Blo99] I. BLOCH, « Fuzzy relative position between objects in image processing : a morphological approach », in : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.7 (1999), p. 657-664.

[BM06] Nilufar BAGHAEI et Antonija MITROVIC, « A Constraint-Based Collaborative Environment for Learning UML Class Diagrams », in : *Intelligent Tutoring Systems*, sous la dir. de Mitsuru IKEDA, Kevin D. ASHLEY et Tak-Wai CHAN, Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, p. 176-186, ISBN : 978-3-540-35160-3.

[Bon+20] Nathalie BONNETON-BOTTÉ et al., « Can tablet apps support the learning of handwriting ? An investigation of learning outcomes in kindergarten classroom », in : *Computers & Education* 151 (2020), p. 103831, ISSN : 0360-1315, DOI : https://doi.org/10.1016/j.compedu.2020.103831, URL : http://www.sciencedirect.com/science/article/pii/S0360131520300336.

[BR05] W. BILLINGSLEY et P. ROBINSON, « Towards an intelligent online textbook for discrete mathematics », in : *Proceedings of the 2005 International Conference on Active Media Technology, 2005. (AMT 2005).* 2005, p. 291-296.

[BS08] Tiffany BARNES et John STAMPER, « Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data », in : *Intelligent Tutoring Systems*, sous la dir. de Beverley P. WOOLF et al., Berlin, Heidelberg : Springer Berlin Heidelberg, 2008, p. 373-382, ISBN : 978-3-540-69132-7.

[BS16] Sahil BHATIA et Rishabh SINGH, *Automated Correction for Syntax Errors in Programming Assignments using Recurrent Neural Networks*, 2016, arXiv : 1603.06129 [cs.PL].

[Car+20] Victor CARBUNE et al., *Fast Multi-language LSTM-based Online Handwriting Recognition*, 2020, arXiv : 1902.10525 [cs.CL].

[Car70] J. R. Carbonell, « AI in CAI : An Artificial-Intelligence Approach to Computer-Assisted Instruction », in : *IEEE Transactions on Man-Machine Systems* 11.*4* (1970), p. 190-202.

[CDR05] G. Costagliola, V. Deufemia et M. Risi, « Sketch Grammars : a formalism for describing and recognizing diagrammatic sketch languages », in : *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, 2005, 1226-1230 Vol. 2.

[CGJ12] Salman Cheema, Sumit Gulwani et Joseph J. LaViola Jr., « QuickDraw : Improving Drawing Experience for Geometric Diagrams », in : *CHI'12, May 5-10, 2012, Austin, Texas, USA*, mai 2012, url : `https://www.microsoft.com/en-us/research/publication/quickdraw-improving-drawing-experience-geometric-diagrams/`.

[CGV02] Cristina Conati, Abigail Gertner et Kurt Vanlehn, « Using Bayesian Networks to Manage Uncertainty in Student Modeling », in : *User Model. User-Adapt. Interact.* 12 (nov. 2002), p. 371-417, doi : `10.1023/A:1021258506583`.

[Cha+14] Kuo Chang et al., « Using mobile devices to enhance the interactive learning for spatial geometry », in : *Interactive Learning Environments* 24 (déc. 2014), p. 1-19, doi : `10.1080/10494820.2014.948458`.

[Che+15] Q. Chen et al., « On-line handwritten flowchart recognition based on logical structure and graph grammar », in : *2015 5th International Conference on Information Science and Technology (ICIST)*, 2015, p. 424-429.

[Chi87] Zelda F. Chickering Arthur W. ; Gamson, « Seven Principles for Good Practice in Undergraduate Education », in : *AAHE Bulletin* 39.*7* (1987), p. 3-7.

[Cho56] N. Chomsky, « Three models for the description of language », in : *IRE Transactions on Information Theory* 2.*3* (1956), p. 113-124.

[CMS00] Albert Corbett, Megan McLaughlin et K. Scarpinatto, « Modeling Student Knowledge : Cognitive Tutors in High School and College », in : *User Model. User-Adapt. Interact.* 10 (juin 2000), p. 81-108, doi : `10.1023/A:1026505626690`.

[Col17] Brice Robert Colby, « A Comparative Literature Review of Intelligent Tutoring Systems from 1992-2015 », in : 2017.

[Cor+10] Albert Corbett et al., « A Cognitive Tutor for Genetics Problem Solving : Learning Gains and Student Modeling », in : *Journal of Educational Computing Research* 42.*2* (2010), p. 219-239, doi : 10.2190/EC.42.2.e, eprint : https://doi.org/10.2190/EC.42.2.e, url : https://doi.org/10.2190/EC.42.2.e.

[Cos+04] G. Costagliola et al., « A Parsing Technique for Sketch Recognition Systems », in : *2004 IEEE Symposium on Visual Languages - Human Centric Computing*, 2004, p. 19-26.

[Cos+11] Gennaro Costagliola et al., « A Sketch-Based System for Teaching Geometry », in : jan. 2011, p. 246-249.

[Cos+93] G. Costagliola et al., « Automatic parser generation for pictorial languages », in : *Proceedings 1993 IEEE Symposium on Visual Languages*, 1993, p. 306-313.

[Coü05] Bertrand Coüasnon, « DMOS, a generic document recognition method : application to table structure analysis in a general and in a specific way », in : *International Journal of Document Analysis and Recognition (IJDAR)* 8 (2005), p. 111-122.

[CP00] G. Costagliola et G. Polese, « Extended positional grammars », in : *Proceeding 2000 IEEE International Symposium on Visual Languages*, 2000, p. 103-110.

[DA13] Adrien Delaye et Eric Anquetil, « HBF49 feature set : A first unified baseline for online symbol recognition », in : *Pattern Recognition* 46.*1* (2013), p. 117-130, issn : 0031-3203, doi : https://doi.org/10.1016/j.patcog.2012.07.015, url : http://www.sciencedirect.com/science/article/pii/S0031320312003317.

[DR20] Vincenzo Deufemia et Michele Risi, « Multi-Domain Recognition of Hand-Drawn Diagrams Using Hierarchical Parsing », in : *Multimodal Technologies and Interaction* 4 (août 2020), p. 52, doi : 10.3390/mti4030052.

[DT05] N. Dalal et B. Triggs, « Histograms of oriented gradients for human detection », in : *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, t. 1, 2005, 886-893 vol. 1.

[Ear14] Yvonne Earnshaw, « Effects of Levels of Instructional Assistance on Learning and Mental Effort in an Intelligent Tutoring System : Proportional Reasoning and Middle School Students », thèse de doct., IFlorida State University, 2014.

[FB93] H. Fahmy et D. Blostein, « A graph grammar programming style for recognition of music notation », in : *Machine Vision and Applications* 6 (1993), p. 83-99.

[FM15] Logan Fiorella et Richard E. Mayer, *Learning as a Generative Activity : Eight Learning Strategies that Promote Understanding*, Cambridge University Press, 2015, doi : 10.1017/CBO9781107707085.

[FM91] Logan Fiorella et Richard E. Mayer, *Active Learning : Creating Excitement in the Classroom*, ASHEERIC Higher Education Report No.1, George Washington University, 1991.

[FO16] Giuseppe Fenza et Francesco Orciuoli, « Building Pedagogical Models by Formal Concept Analysis », in : *Intelligent Tutoring Systems*, sous la dir. d'Alessandro Micarelli, John Stamper et Kitty Panourgia, Cham : Springer International Publishing, 2016, p. 144-153, isbn : 978-3-319-39583-8.

[Fon+20] Ludovic Font et al., « Automating the Generation of High School Geometry Proofs using Prolog in an Educational Context », in : *Electronic Proceedings in Theoretical Computer Science* 313 (fév. 2020), p. 1-16, issn : 2075-2180, doi : 10.4204/eptcs.313.1, url : http://dx.doi.org/10.4204/EPTCS.313.1.

[For10] Kenneth D. Forbus, « CogSketch : Sketch Understanding for Cognitive Science Research and for Education », in : *Spatial Cognition VII*, sous la dir. de Christoph Hölscher et al., Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, p. 4-4, isbn : 978-3-642-14749-4.

[Fos+09] Davide Fossati et al., « I learn from you, you learn from me : How to make iList . . . », in : jan. 2009, p. 491-498, doi : 10.3233/978-1-60750-028-5-491.

[FRG18] Ludovic Font, Philippe R. Richard et Michel Gagnon, « Improving QED-Tutrix by Automating the Generation of Proofs », in : *Electronic Proceedings in Theoretical Computer Science* 267 (mars 2018), p. 38-58, issn : 2075-2180,

DOI : 10.4204/eptcs.267.3, URL : http://dx.doi.org/10.4204/EPTCS.267.3.

[GGC09] Jaime GÁLVEZ, Eduardo GUZMÁN et Ricardo CONEJO, « A blended E-learning experience in a course of object oriented programming fundamentals », in : *Knowledge-Based Systems* 22.*4* (2009), Artificial Intelligence (AI) in Blended Learning, p. 279-286, ISSN : 0950-7051, DOI : https://doi.org/10.1016/j.knosys.2009.01.004, URL : http://www.sciencedirect.com/science/article/pii/S0950705109000203.

[Gho12] Achraf GHORBEL, « Interactive interpretation of structured documents : application to the retro-conversion of handwritten architectural plans », Theses, INSA de Rennes, déc. 2012, URL : https://tel.archives-ouvertes.fr/tel-00788832.

[Gin+14] Blandine GINON et al., « Adding Epiphytic Assistance Systems in Learning Applications Using the SEPIA System », in : *Open Learning and Teaching in Educational Communities*, sous la dir. de Christoph RENSING et al., Cham : Springer International Publishing, 2014, p. 138-151.

[GKT11] Sumit GULWANI, Vijay Anand KORTHIKANTI et Ashish TIWARI, « Synthesizing Geometry Constructions », in : *SIGPLAN Not.* 46.*6* (juin 2011), p. 50-61, ISSN : 0362-1340, DOI : 10.1145/1993316.1993505, URL : http://doi.acm.org/10.1145/1993316.1993505.

[GNT04] Malik GHALLAB, Dana NAU et Paolo TRAVERSO, *Automated planning. Theory & practice*, mai 2004, ISBN : 978-1-55860-856-6.

[Gob+10] Janice D. GOBERT et al., « The Science Assistments Project : Scaffolding Scientific Inquiry Skills », in : *Intelligent Tutoring Systems*, sous la dir. de Vincent ALEVEN, Judy KAY et Jack MOSTOW, Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, p. 445-445, ISBN : 978-3-642-13437-1.

[GSA17] Nathalie GIRARD, Damien SIMONNET et Eric ANQUETIL, « IntuiScript a new digital notebook for learning writing in elementary schools : 1st observations », in : *18th International Graphonomics Society Conference (IGS2017)*, Proceedings of IGS 2017, Gaeta, Italy, juin 2017, p. 201-204, URL : https://hal.inria.fr/hal-01548200.

[Gul10] Sumit Gulwani, « Dimensions in Program Synthesis », in : *Proceedings of the 12th International ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming*, PPDP '10, Hagenberg, Austria : ACM, 2010, p. 13-24, isbn : 978-1-4503-0132-9, doi : 10.1145/1836089.1836091, url : http://doi.acm.org/10.1145/1836089.1836091.

[Gup+17] Rahul Gupta et al., « DeepFix : Fixing Common C Language Errors by Deep Learning », in : *AAAI*, 2017.

[HD05] Tracy Hammond et Randall Davis, « LADDER, a sketching language for user interface developers », in : *Computers & Graphics* 29.*4* (2005), p. 518-532, issn : 0097-8493, doi : https://doi.org/10.1016/j.cag.2005.05.005, url : http://www.sciencedirect.com/science/article/pii/S0097849305000865.

[Hef+06] Neil Heffernan et al., « The ASSISTment Builder : Towards an Analysis of Cost Effectiveness of ITS Creation. », in : jan. 2006, p. 515-520.

[HH14] Neil Heffernan et Cristina Heffernan, « The ASSISTments Ecosystem : Building a Platform that Brings Scientists and Teachers Together for Minimally Invasive Research on Human Learning and Teaching », in : *International Journal of Artificial Intelligence in Education* 24 (déc. 2014), doi : 10.1007/s40593-014-0024-x.

[Hir+09a] C Hirashima et al., « Constraint-based Design Critic for Flat-pack Furniture Design », in : *Proceedings of the 17th International Conference on Computers in Education, ICCE 2009* 19 (jan. 2009).

[Hir+09b] C Hirashima et al., « Evaluation of a constraint-based homework assistance system for logic programming », in : (jan. 2009).

[HMM09] J. Holland, Antonija Mitrovic et Brent Martin, « J-Latte : a Constraint-Based Tutor for Java », in : (jan. 2009).

[HTL15] L. de las Heras, O. R. Terrades et J. Lladós, « Attributed Graph Grammar for floor plan analysis », in : *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, 2015, p. 726-730.

[Hub15] Sarka Hubackova, « History and Perspectives of Elearning », in : *Procedia - Social and Behavioral Sciences* 191 (juin 2015), p. 1187-1190, doi : 10.1016/j.sbspro.2015.04.594.

[IHB14] Barry W. Peddycord III, Andrew Hicks et Tiffany Barnes, « Generating Hints for Programming Problems Using Intermediate Output », in : *Proceedings of the 7th International Conference on Educational Data Mining, EDM 2014, London, UK, July 4-7, 2014*, sous la dir. de John C. Stamper et al., International Educational Data Mining Society (IEDMS), 2014, p. 92-98, URL : http://www.educationaldatamining.org/EDM2014/uploads/procs2014/longpapers/92%5C_EDM-2014-Full.pdf.

[Iso+14] S. Isotani et al., « Interactive Geometry Goes Mobile with GeoTouch », in : *2014 IEEE 14th International Conference on Advanced Learning Technologies*, 2014, p. 181-185.

[Jaq+13] Patricia A. Jaques et al., « Rule-based expert systems to support step-by-step guidance in algebraic problem solving : The case of the tutor PAT2Math », in : *Expert Systems with Applications* 40.*14* (2013), p. 5456-5465, ISSN : 0957-4174, DOI : https://doi.org/10.1016/j.eswa.2013.04.004, URL : http://www.sciencedirect.com/science/article/pii/S0957417413002418.

[JG95] J. A. P. Jorge et E. P. Glinert, « Online parsing of visual languages using adjacency grammars », in : *Proceedings of Symposium on Visual Languages*, 1995, p. 250-257.

[Jin+12] Wei Jin et al., « Program Representation for Automatic Hint Generation for a Data-Driven Novice Programming Tutor », in : *Intelligent Tutoring Systems*, sous la dir. de Stefano A. Cerri et al., Berlin, Heidelberg : Springer Berlin Heidelberg, 2012, p. 304-309, ISBN : 978-3-642-30950-2.

[Jin+14] Wei Jin et al., « Evaluation of Guided-Planning and Assisted-Coding with Task Relevant Dynamic Hinting », in : *Intelligent Tutoring Systems*, 2014.

[Jul+17] Frank Julca-Aguilar et al., *A General Framework for the Recognition of Online Handwritten Graphics*, 2017, arXiv : 1709.06389 [cs.CV].

[KAO14] Hokuto Kagaya, Kiyoharu Aizawa et Makoto Ogawa, « Food Detection and Recognition Using Convolutional Neural Network », in : nov. 2014, DOI : 10.13140/2.1.3082.1120.

[KD96] Avraham N. Kluger et Angelo Denisi, « The effects of feedback interventions on performance : A historical review, a meta-analysis, and a preliminary feedback intervention theory », in : *Psychological Bulletin* (1996), p. 254-284.

[Koc16] Uwe Kockemann, « Constraint-based Methods for Human-aware Planning », in : 2016.

[Koe+04] Kenneth Koedinger et al., « Opening the Door to Non-programmers : Authoring Intelligent Tutor Behavior by Demonstration », in : t. 3220, août 2004, p. 162-174, doi : 10.1007/978-3-540-30139-4_16.

[Kum06] A. Kumar, « Using Enhanced Concept Map for Student Modeling in Programming Tutors », in : *FLAIRS Conference*, 2006.

[Lab02] Colette Laborde, « Integration of Technology in the Design of Geometry Tasks with Cabri-Geometry », in : *I. J. Computers for Math. Learning* 6.*3* (2002), p. 283-317, doi : 10.1023/A\%3A1013309728825, url : https://doi.org/10.1023/A%5C%3A1013309728825.

[Lem+13] Aurélie Lemaitre et al., « Interest of syntactic knowledge for on-line flowchart recognition », in : *Graphics Recognition New Trends and Challenges*, sous la dir. d'Young-Bin Kwon et Jean-Marc Ogier, t. 7423, LNCS, Springer, déc. 2013, p. 89-98, doi : 10.1007/978-3-642-36824-0_9, url : https://hal.inria.fr/hal-00854450.

[LMZ16] Dejan Lavbič, Tadej Matek et Aljaz Zrnec, « Recommender system for learning SQL using hints », in : *Interactive Learning Environments* (oct. 2016), doi : 10.1080/10494820.2016.1244084.

[Low04] David Lowe, « Distinctive Image Features from Scale-Invariant Keypoints », in : *International Journal of Computer Vision* 60 (nov. 2004), p. 91-110, doi : 10.1023/B%3AVISI.0000029664.99615.94.

[MA09] Sébastien Macé et Eric Anquetil, « Eager interpretation of on-line hand-drawn structured documents : The DALI methodology », in : *Pattern Recognition* 42.*12* (2009), New Frontiers in Handwriting Recognition, p. 3202-3214, issn : 0031-3203, doi : https://doi.org/10.1016/j.patcog.2008.10.018, url : http://www.sciencedirect.com/science/article/pii/S0031320308004482.

[MA19] A. Marouf et S. S. Abu-Naser, « Intelligent Tutoring System for Teaching Computer Science I in Al-Azhar University, Gaza », in : *International Journal of Academic and Applied Research (IJAAR)* 3.*3* (2019), p. 31-53, issn : 0020-7373, doi : https://doi.org/10.1016/S0020-7373(74)

80005-2, URL : http://www.sciencedirect.com/science/article/pii/S0020737374800052.

[MAH00] Jennifer MANKOFF, Gregory D ABOWD et Scott E HUDSON, « OOPS : a toolkit supporting mediation techniques for resolving ambiguity in recognition-based interfaces », in : *Computers & Graphics* 24.*6* (2000), Calligraphic Interfaces : towards a new generation of interactive systems, p. 819-834, ISSN : 0097-8493, DOI : https://doi.org/10.1016/S0097-8493(00)00085-6, URL : http://www.sciencedirect.com/science/article/pii/S0097849300000856.

[Mar94] K. MARRIOTT, « Constraint multiset grammars », in : *Proceedings of 1994 IEEE Symposium on Visual Languages*, 1994, p. 118-125.

[Mas+10] J. MAS et al., « A syntactic approach based on distortion-tolerant Adjacency Grammars and a spatial-directed parser to interpret sketched diagrams », in : *Pattern Recognition* 43.*12* (2010), p. 4148-4164, ISSN : 0031-3203, DOI : https://doi.org/10.1016/j.patcog.2010.07.003, URL : http://www.sciencedirect.com/science/article/pii/S0031320310003420.

[McL+15] Bruce M. MCLAREN et al., « Worked Examples are More Efficient for Learning than High-Assistance Instructional Software », in : *Artificial Intelligence in Education*, sous la dir. de Cristina CONATI et al., Cham : Springer International Publishing, 2015, p. 710-713, ISBN : 978-3-319-19773-9.

[MH69] John MCCARTHY et Patrick J. HAYES, « Some Philosophical Problems from the Standpoint of Artificial Intelligence », in : *Machine Intelligence 4*, sous la dir. de B. MELTZER et D. MICHIE, reprinted in McC90, Edinburgh University Press, 1969, p. 463-502.

[Mit+11] A. MITROVIC et al., « Thermo-Tutor : An Intelligent Tutoring System for thermodynamics », in : *2011 IEEE Global Engineering Education Conference (EDUCON)*, 2011, p. 378-385.

[Mit10] Antonija MITROVIC, « Modeling Domains and Students with Constraint-Based Modeling », in : *Advances in Intelligent Tutoring Systems*, sous la dir. de Roger NKAMBOU, Jacqueline BOURDEAU et Riichiro MIZOGUCHI, Berlin, Heidelberg : Springer Berlin Heidelberg, 2010, p. 63-80, ISBN : 978-3-642-14363-2, DOI : 10.1007/978-3-642-14363-2_4, URL : https://doi.org/10.1007/978-3-642-14363-2_4.

[Mit12] Antonija MITROVIC, « Fifteen years of constraint-based tutors : What we have achieved and where we are going », in : *User Modeling and User-Adapted Interaction* 22 (avr. 2012), p. 39-72, DOI : 10.1007/s11257-011-9105-9.

[MJ71] Michael S. MATELL et Jacob JACOBY, « Is There an Optimal Number of Alternatives for Likert Scale Items ? Study I : Reliability and Validity », in : *Educational and Psychological Measurement* 31.*3* (1971), p. 657-674, DOI : 10.1177/001316447103100307, eprint : https://doi.org/10.1177/001316447103100307, URL : https://doi.org/10.1177/001316447103100307.

[Mor+17] Felipe de MORAIS et al., « The Use of Handwriting Input in Math Tutoring Systems : An Use Case with PAT2Math », in : juill. 2017, p. 44-46, DOI : 10.1109/ICALT.2017.142.

[MSL06] Joan MAS, Gemma SANCHEZ et Josep LLADOS, « An Incremental Parser to Recognize Diagram Symbols and Gestures Represented by Adjacency Grammars », in : *Graphics Recognition. Ten Years Review and Future Perspectives*, sous la dir. de Wenyin LIU et Josep LLADOS, Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, p. 243-254, ISBN : 978-3-540-34712-5.

[MV05] Noboru MATSUDA et Kurt VANLEHN, « Advanced Geometry Tutor : An intelligent tutor that teaches proof-writing with construction. », in : jan. 2005, p. 443-450.

[MW13] Carsten Miller MATTHIAS EHMANN Michael Gerhauser et Alfred WASSERMANN, « SketchGeometry and JXGraph : dynamic geometry for mobile devices », in : (2013).

[Nas16] Samy S. Abu NASER, « ITSB : An Intelligent Tutoring System Authoring Tool », in : 2016.

[Ngo+17] Anh Khoi NGO HO et al., « A multi-one-class dynamic classifier for adaptive digitization of document streams », in : *International Journal on Document Analysis and Recognition* (mai 2017), p. 1-18, DOI : 10.1007/s10032-017-0286-6, URL : https://hal.archives-ouvertes.fr/hal-01525831.

[NIE93] JAKOB NIELSEN, « Chapter 5 - Usability Heuristics », in : *Usability Engineering*, sous la dir. de JAKOB NIELSEN, San Diego : Morgan Kaufmann, 1993, p. 115-163, ISBN : 978-0-12-518406-9, DOI : https://doi.org/10.

1016/B978-0-08-052029-2.50008-5, URL : http://www.sciencedirect.com/science/article/pii/B9780080520292500085.

[NMB10] Roger NKAMBOU, Riichiro MIZOGUCHI et Jacqueline BOURDEAU, *Advances in Intelligent Tutoring Systems*, t. 308, jan. 2010, DOI : 10.1007/978-3-642-14363-2.

[Ohl94] Stellan OHLSSON, « Constraint-Based Student Modeling », in : *Student Modelling : The Key to Individualized Knowledge-Based Instruction*, sous la dir. de Jim E. GREER et Gordon I. MCCALLA, Berlin, Heidelberg : Springer Berlin Heidelberg, 1994, p. 167-189, ISBN : 978-3-662-03037-0.

[Ohl96] Stellan OHLSSON, « Learning from error and the design of task environments », in : *International Journal of Educational Research* 25.5 (1996), p. 419-448, ISSN : 0883-0355, DOI : https://doi.org/10.1016/S0883-0355(97)81236-0, URL : http://www.sciencedirect.com/science/article/pii/S0883035597812360.

[PH08] Brandon PAULSON et Tracy HAMMOND, « PaleoSketch : Accurate primitive sketch recognition and beautification », in : jan. 2008, p. 1-10, DOI : 10.1145/1378773.1378775.

[PMA10] Mathieu PÉCOT, Sébastien MACÉ et Eric ANQUETIL, « Interprétation interactive de plans d'architecture composés à main levée », in : *Colloque International Francophone sur l'Écrit et le Document, Hammamet, Tunisie*, 2010.

[PP16] Melissa M. PATCHAN et Cynthia S. PURANIK, « Using tablet computers to teach preschool children to write letters : Exploring the impact of extrinsic and intrinsic feedback », in : *Computers & Education* 102 (2016), p. 128-137, ISSN : 0360-1315, DOI : https://doi.org/10.1016/j.compedu.2016.07.007, URL : http://www.sciencedirect.com/science/article/pii/S0360131516301439.

[PR06] Patricia Gerent PETRY et Marta Costa ROSATELLI, « AlgoLC : A Learning Companion System for Teaching and Learning Algorithms », in : *Intelligent Tutoring Systems*, sous la dir. de Mitsuru IKEDA, Kevin D. ASHLEY et Tak-Wai CHAN, Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, p. 775-777, ISBN : 978-3-540-35160-3.

[Py01] Dominique Py, *Environnements interactifs d'apprentissage et démonstration en géométrie*, Habilitation à diriger des recherches de l'université de Rennes I, juillet 2001., 2001, URL : https://telearn.archives-ouvertes.fr/hal-00190202.

[RBE00] Jean-Yves Ramel, Guillaume Boissier et Hubert Emptoz, « A Structural Representation Adapted to Handwritten Symbol Recognition », in : *Graphics Recognition Recent Advances*, sous la dir. d'Atul K. Chhabra et Dov Dori, Berlin, Heidelberg : Springer Berlin Heidelberg, 2000, p. 228-237, ISBN : 978-3-540-40953-3.

[RM87] D. E. Rumelhart et J. L. McClelland, « Learning Internal Representations by Error Propagation », in : *Parallel Distributed Processing : Explorations in the Microstructure of Cognition : Foundations*, 1987, p. 318-362.

[SAT10] Royati Abdul Saha, Ahmad Fauzi Mohd Ayub et Rohani Ahmad Tarmizi, « The Effects of GeoGebra on Mathematics Achievement : Enlightening Coordinate Geometry Learning », in : *Procedia - Social and Behavioral Sciences* 8 (2010), International Conference on Mathematics Education Research 2010 (ICMER 2010), p. 686-693, ISSN : 1877-0428, DOI : https://doi.org/10.1016/j.sbspro.2010.12.095, URL : http://www.sciencedirect.com/science/article/pii/S1877042810022007.

[Sch00] Daniel Scher, « Lifting the Curtain : The Evolution of The Geometer's Sketchpad », in : *The mathematics Educator* 10.*2* (2000), p. 42-48.

[Sel74] John A. Self, « Student models in computer-aided instruction », in : *International Journal of Man-Machine Studies* 6.*2* (1974), p. 261-276, ISSN : 0020-7373, DOI : https://doi.org/10.1016/S0020-7373(74)80005-2, URL : http://www.sciencedirect.com/science/article/pii/S0020737374800052.

[Sha70] A. Shaw, « Parsing of Graph-Representable Pictures », in : *J. ACM* 17 (1970), p. 453-481.

[SHM14] Rahul Singhal, Martin Henz et Kevin McGee, « Automated Generation of High School Geometric Questions Involving Implicit Construction », in : *Proceedings of the 6th International Conference on Computer Supported Education - Volume 1*, CSEDU 2014, Barcelona, Spain, 2014, p. 467-472, ISBN :

978-989-758-020-8, DOI : 10.5220/0004947904670472, URL : http://dx.doi.org/10.5220/0004947904670472.

[Sim+18] Damien SIMONNET et al., « Evaluation of Children Cursive Handwritten Words for e-Education », in : *Pattern Recognition Letters* (juill. 2018), DOI : 10.1016/j.patrec.2018.07.021.

[SKV16] Ravi Kiran SARVADEVABHATLA, Jogendra KUNDU et Babu R. VENKATESH, *Enabling My Robot To Play Pictionary : Recurrent Neural Networks For Sketch Recognition*, 2016, arXiv : 1608.03369 [cs.CV].

[SM02] Pramuditha SURAWEERA et Antonija MITROVIC, « KERMIT : A Constraint-Based Tutor for Database Modeling », in : *Intelligent Tutoring Systems*, sous la dir. de Stefano A. CERRI, Guy GOUARDÈRES et Fàbio PARAGUAÇU, Berlin, Heidelberg : Springer Berlin Heidelberg, 2002, p. 377-387, ISBN : 978-3-540-47987-1.

[TA16] Nyamen TATO et A Tymiak ADRIENNE, « Développement d'un système tutoriel intelligent pour l'apprentissage du raisonnement logique », thèse de doct., Université du Québec à Montréal, Maîtrise en informatique, 2016.

[TBH15] Paul TAELE, Laura BARRETO et Tracy HAMMOND, « Maestoso : An Intelligent Educational Sketching Tool for Learning Music Theory », in : *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, Austin, Texas : AAAI Press, 2015, p. 3999-4005, ISBN : 0262511290.

[TS85] T. TAKAGI et M. SUGENO, « Fuzzy identification of systems and its applications to modeling and control », in : *IEEE Transactions on Systems, Man, and Cybernetics* SMC-15.*1* (1985), p. 116-132.

[Uhr69] Leonard UHR, « Teaching Machine Programs That Generate Problems as a Function of Interaction with Students », in : *Proceedings of the 1969 24th National Conference*, ACM '69, New York, NY, USA : Association for Computing Machinery, 1969, p. 125-134, ISBN : 9781450374934, DOI : 10.1145/800195.805924, URL : https://doi.org/10.1145/800195.805924.

[Wan05] Lipo WANG, « Support Vector Machines : Theory and Applications », in : *Studies in fuzziness and soft computing, v 177* 302 (jan. 2005).

[WCZ18] X. Wang, X. Chen et Z. Zha, « Sketchpointnet : A Compact Network for Robust Sketch Recognition », in : *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, p. 2994-2998.

[WZY07] Liu Wenyin, Wan Zhang et Luo Yan, « An interactive example-driven approach to graphics recognition in engineering drawings », in : *IJDAR* 9 (fév. 2007), p. 13-29, doi : 10.1007/s10032-006-0025-x.

[XJB19] Peng Xu, Chaitanya K. Joshi et Xavier Bresson, *Multi-Graph Transformer for Free-Hand Sketch Recognition*, 2019, arXiv : 1912.11258 [cs.CV].

[Yan+18] Jihong Yan et al., « A retrospective of knowledge graphs », in : *Frontiers of Computer Science* 12.*1* (fév. 2018), p. 55-74, issn : 2095-2236, doi : 10.1007/s11704-016-5228-9, url : https://doi.org/10.1007/s11704-016-5228-9.

[You67] Daniel H. Younger, « Recognition and parsing of context-free languages in time n3 », in : *Information and Control* 10.*2* (1967), p. 189-208, issn : 0019-9958, doi : https://doi.org/10.1016/S0019-9958(67)80007-X, url : http://www.sciencedirect.com/science/article/pii/S001999586780007X.

[Yu+15] Qian Yu et al., *Sketch-a-Net that Beats Humans*, 2015, arXiv : 1501.07873 [cs.CV].

[Zha+20] Xingyuan Zhang et al., « A Hybrid convolutional neural network for sketch recognition », in : *Pattern Recognition Letters* 130 (2020), Image/Video Understanding and Analysis (IUVA), p. 73-82, issn : 0167-8655, doi : https://doi.org/10.1016/j.patrec.2019.01.006, url : http://www.sciencedirect.com/science/article/pii/S0167865519300078.

[ZMV13] Richard Zanibbi, Harold Mouchère et Christian Viard-Gaudin, « Evaluating Structural Pattern Recognition for Handwritten Math via Primitive Label Graphs », in : t. 8658, fév. 2013, p. 865817-865817, doi : 10.1117/12.2008409.

INSA de RENNES
Service des Formations

AVIS DU JURY SUR LA REPRODUCTION DE LA THESE SOUTENUE

Titre de la thèse:
Conception d'un système tutoriel intelligent orienté stylet pour l'apprentissage de la géométrie basé une interprétation à la volée de la production manuscrite de figures

Nom Prénom de l'auteur : KRICHEN OMAR

Membres du jury :
- Madame JEAN-DAUBIAS Stéphanie
- Monsieur MOUCHERE Harold
- Madame PY Dominique
- Monsieur ANQUETIL Eric
- Monsieur RAMEL Jean-Yves
- Madame EGLIN Véronique

Président du jury :

Date de la soutenance : 04 Décembre 2020

Reproduction de la these soutenue

☒ Thèse pouvant être reproduite en l'état
☐ Thèse pouvant être reproduite après corrections suggérées

Fait à Rennes, le 04 Décembre 2020

Signature du président de jury

Le Directeur,

M'hamed DRISSI

DOCTORAT BRETAGNE LOIRE / MATHSTIC

INSA RENNES

Titre : Conception d'un système tutoriel intelligent orienté stylet pour l'apprentissage de la géométrie basé sur une interprétation à la volée de la production manuscrite de figures

Mot clés : Systèmes tutoriels intelligents, interprétation à la volée de documents manuscrits, planification

Résumé : Cette thèse s'inscrit dans le cadre du projet national « e-Fran » dénommé ACTIF et porte sur la conception du système tutoriel intelligent IntuiGeo pour l'apprentissage de la géométrie au collège sur tablette orientée stylet. Les contributions de cette thèse s'inscrivent dans deux axes. Le premier porte sur la conception d'un moteur de reconnaissance permettant l'interprétation à la volée de figures géométriques manuscrites. Il est basé sur un formalisme grammatical générique, GMC-PC (Grammaire Multi-ensembles à Contraintes Pilotée par le Contexte). Le principal challenge étant de gérer la complexité de l'analyse en temps-réel, la première contribution cette thèse a consisté à étendre ce formalisme, en conservant son aspect générique. Le deuxième axe adresse l'aspect tutoriel du système. Nous définissons un mode auteur qui permet au tuteur de générer des exercices de construction à partir d'une solution dessinée par l'enseignant. La connaissance spécifique au problème est représentée par un graphe de connaissance. Cette modélisation permet au tuteur de s'affranchir de la procédure suivie par l'enseignant et d'évaluer la production de l'élève, en temps-réel, quel que soit la stratégie suivie. Nous définissons de plus un module expert, basé sur un environnement de planification, capable de synthétiser des stratégies de résolution des problèmes. Le système tutoriel est capable de générer des feedbacks de correction et de guidage adaptés à l'état de l'avancement de l'élève. Les résultats des expérimentations en classe démontrent l'impact pédagogique positif du système sur la performance des élèves, notamment en termes de transfert d'apprentissage entre support numérique et papier.

Title: Design of an intelligent tutoring system for geometry learning based on on-the-fly interpretation of hand-drawn figures production

Keywords: Intelligent tutoring systems, on the fly interpretation of hand-drawn documents, planning

Abstract: This PhD is in the context of the « e-Fran » national project called ACTIF and deals with the design of the pen-based intelligent tutoring system IntuiGeo, for geometry learning in middle school. The contribution of this work are grouped into two axes. The first axis focused on the design of a recognition engine capable of on the fly interpretation of Han-drawn geometrical figures. It is based on a generic grammatical formalism, CD-CMG (Context Driven Constraints Multiset Grammar). The challenge being to manage the complexity of the real-time analysis process, the first contribution of this work consisted in extending the formalism, without losing its generic aspect. The second axis of this work addresses the tutorial aspect of our system. We define au author mode where the tutor is able to generate construction exercises from a solution example drawn by the teacher. The problem specific knowledge is represented by a knowledge graph. This representation enables the tutor to consider all possible resolution strategies, and to evaluate the pupil's production in real-time. Furthermore, we define an expert module, based on a dynamic planning environment, capable of synthesizing resolution strategies. The tutoring system is able to generate guidance and corrective feedbacks that are adapted to the pupil's resolution state. The results of our experiment conducted in class demonstrate the positive pedagogical impact of the system on the pupils performance, especially in terms of learning transferability between the digital and traditional support.