Un moteur d'appariement et transformation de sous-arbres pour la conception interactive de scénarios d'analyse d'images de documents

Pascal Bourquin, Jean-Yves Ramel

Laboratoire Informatique de Tours- Université François-Rabelais
64 avenue Portalis 37200 Tours, France
pascal.bourquin@univ-tours.fr, jean-yves.ramel@univ-tours.fr

RÉSUMÉ. *Cet article décrit un nouveau moteur d'extraction d'éléments de contenu et d'analyse de structures de pages numérisées. Cette nouvelle méthode se base sur un mécanisme original d'appariement séquentiel et de transformation de sous-arbres. Les sous-arbres permettent la description des structures à localiser dans l'état courant de l'interprétation du document représenté également par une structure arborescente. Un algorithme de recherche d'appariements sous contraintes de sous-arbres pouvant être exploité de manière incrémental permet la mise en place interactive de scénarios d'analyse cherchant à répondre aux besoins spécifiques de chaque usager. Ainsi, l'originalité de notre approche réside dans l'opportunité que nous offrons aux utilisateurs de pouvoir construire de manière interactive des scénarios d'analyse incrémentale d'images à moindre coût. Le but est de laisser l'utilisateur concevoir comme bon lui semble sa chaine de traitement en faisant évoluer une représentation des images sous forme d'arbre d'éléments de contenu de manière progressive.*

MOTS-CLES : segmentation d'éléments de contenu, analyse de la mise en page, stratégie d'analyse, appariement sous contrainte de sous-arbres

ABSTRACT: This paper describes a new framework dedicated to layout analysis and content extraction in digitized documents. This new method is based on an original sub-tree transformation and matching algorithm that includes a constraint verification step. The sub-tree structures represent the elements of content to be extracted as well as the current state of the interpretation process. The sub-tree transformation and matching algorithm can be used incrementally in order to build adaptive processing chains making the tree corresponding to the whole document evolving. The originality of the proposed framework comes from the possibility provided to the users to generate interactively, and as they wants, many different image analysis scenarios by just applying sequentially some specific transformations on a tree structure that represents the layout of the document images

CIFED 2014, pp. 309–324, Nancy, 18-21 mars 2014

1. Introduction

Nous présentons dans cet article une partie des travaux effectués dans le cadre du projet Paradiit financé par deux Google awards successifs et mené en collaboration avec le Centre d'Etudes Supérieures de la Renaissance (CESR) de Tours. Notre projet vise à mettre à disposition un ensemble d'outils interactifs et de nouvelles bases de connaissances permettant une meilleure analyse, transcription et indexation des ouvrages anciens imprimés. Pour cela, nous pensons qu'il est nécessaire de produire des outils exploitant des méta-données d'indexation aussi bien perceptuelles (orientées images) que sémantiques (contenu historique et notice bibliographique).

Dans ce cadre, cet article décrit un nouveau moteur d'extraction d'éléments de contenu et d'analyse de structures de pages numérisées. Ce moteur a été implémenté au sein de la dernière version du logiciel Agora disponible au téléchargement sur le site du projet Paradiit [Paradiit2013]. Cette nouvelle méthode d'extraction de structures et d'éléments de contenu se base sur un mécanisme d'appariement séquentiel de sous-arbres. Les sous-arbres permettent la description des structures à localiser dans l'état courant de l'interprétation du document représenté également par une structure arborescente. Un algorithme original de recherche d'appariements sous contraintes pouvant être exploité de manière incrémentale permet la mise en place interactive de scénarios d'analyse permettant ainsi de répondre aux besoins spécifiques de chaque usager.

Visant toujours un objectif de généricité, l'architecture de la nouvelle version d'Agora autorise, comme les précédentes [Ramel2007], la mise en place interactive de scénarios d'analyse des contenus. En fonction des spécificités des images à traiter et de ses besoins (localisation des lettrines, des titres de section, …) et à l'aide d'interfaces conviviales, l'utilisateur construit des scénarios permettant d'étiqueter, de fusionner et de supprimer les structures arborescentes (sous-arbres) correspondant à des motifs spécifiés de manière interactive. Il localise ainsi les entités qui l'intéressent en ignorant les autres régions de l'image (branches de l'arbre). Les scénarios élaborés peuvent ensuite être sauvegardés, modifiés et appliqués sur différentes images lors de traitements par lots.

Tout d'abord, en section 2, nous analysons les principales approches existantes pour l'analyse de structures, et montrons que la généricité d'un système ne peut être obtenue que par interaction avec un utilisateur pour guider l'analyse. Contrairement, aux approches préalablement développées, nous proposons de permettre à l'utilisateur de définir l'ordre et les critères d'extraction des éléments pertinents sur la base de ses connaissances sur les différents lots d'images à traiter (sans forcément encoder un réel modèle de document) plutôt que d'attendre que le système génère des erreurs difficiles à corriger pour intervenir.. Dans les sections 3 à 5, nous présentons la modélisation des données et l'architecture requises pour mettre en œuvre ce mode d'interaction autorisant la définition de processus d'analyse adaptés aux caractéristiques des images à traiter. Finalement, en section 6, nous illustrons les

capacités d'un tel système par quelques expériences que nous avons menées avant de conclure sur l'intérêt de ce travail.

2. Analyse de structures, un problème loin d'être résolu…

Selon [Michard2000], le concept de balisage structurel est assez simple à comprendre et à appréhender. Il suppose que tout document textuel est construit selon une structure, qui peut être reconnue par les lecteurs humains grâce à des marques typographiques, des conventions de mise en page et des connaissances 'pragmatiques' ou culturelles relatives aux informations génériques qu'est susceptible de contenir tout document particulier appartenant à une certaine catégorie de documents.

De nombreux travaux ont cherché à exploiter certaines de ces informations pour mettre en place des systèmes automatiques d'analyse de structures notamment durant les années 1995 à 2005 mais aucun ne semble avoir abouti à un dispositif opérationnel suffisamment générique. Ces méthodes automatiques peuvent être classées en différentes catégories. Une première classification peut être faite selon l'objectif visé qui peut concerner :

- soit la segmentation de l'image à proprement dite c'est-à-dire l'extraction de blocs homogènes dans l'image ; on parle alors de structure physique,

- soit l'extraction, l'étiquetage et l'analyse de l'enchainement entre éléments d'information caractérisés par le rôle qu'ils jouent dans le document ; on parle alors de la structure logique du document.

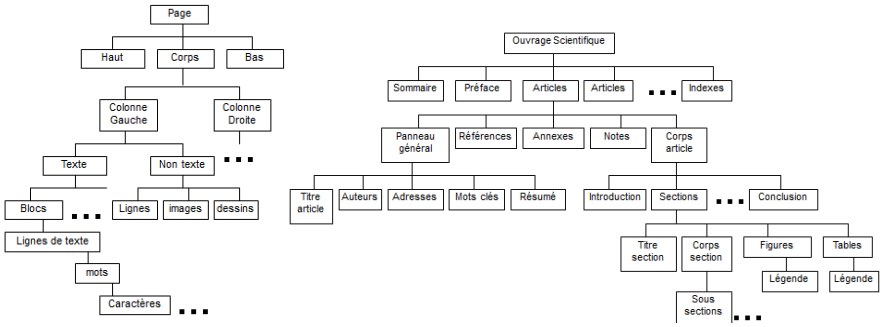
Algorithmes et méthodes automatiques pour l'analyse de structures

La structure physique correspond à la structure perceptible telle qu'elle apparaît visuellement. Elle décrit l'organisation du document, en termes d'objets graphiques (signes typographiques, mots, lignes, blocs, zones graphiques) et des relations entre ces objets (décomposition hiérarchique, positions absolues et relatives dans la page).

Comme le montre la figure 1a, une structure arborescente parait tout à fait adaptée pour représenter la structure physique correspondant à une image de document, chaque nœud de l'arbre correspond à un élément physique plus ou moins élémentaire associé à une région de l'image que l'on va alors chercher à extraire automatiquement.

Il est possible de classifier les techniques d'analyse de structures physique en fonction des méthodes de traitement d'images exploitées. Parmi les méthodes dites ascendantes, on trouve les méthodes se basant sur des techniques de filtrages morphologiques ou différentiels. L'idée est d'utiliser des filtres permettant d'agglomérer les variations d'intensité périodiquement produites par les contours des caractères puis de rechercher des alignements horizontaux pour les lignes de texte. Ce type de méthodes a été beaucoup mis en œuvre pour détecter les mots, lignes ou

paragraphes de texte dans les images de documents anciens manuscrits ou imprimés [Alaei2011] [Lebourgeois03].



(a) (b)

Figure 1 : Structure physique (a) et logique (b) d'un document

D'autres méthodes se basent sur l'ensemble des composantes connexes obtenues après binarisation qu'elles cherchent à classifier ou agglomérer pour constituer des éléments de plus haut niveau Sur la figure 2, on peut constater que la taille, la proximité et la position relative des composantes connexes peuvent être utilisées pour analyser la structure physique d'une page. Les rectangles circonscrits aux composantes connexes se chevauchent fréquemment dans les zones graphiques et rarement dans les textes. De plus, les éléments graphiques correspondent la plupart du temps aux composantes connexes dont les dimensions dépassent un seuil facilement définissable tandis qu'un bloc de texte est un ensemble de petites composantes connexes « proches » [OGorman93].

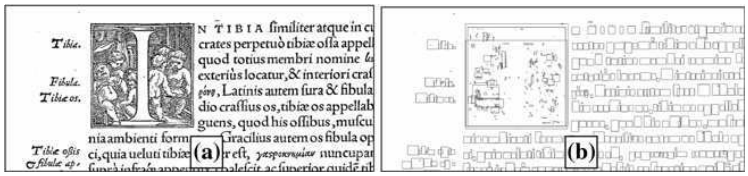


Figure 2 : Composantes connexes et segmentation

La variété des informations apportées par les composantes connexes, nous laisse voir qu'il semble impensable de ne pas chercher à exploiter ce type d'approche dans un système d'analyse de document. Le problème à résoudre provient de la non généricité des paramétrages qui oblige à revoir le système dès qu'un nouveau type ou lot d'images doit être traité. Si des études statistiques peuvent parfois permettre d'automatiser la sélection de ces seuils, ces méthodes nécessitent d'avoir des données d'apprentissage étiquetées en nombre ce qui est bien souvent impossible.

Les approches descendantes tentent de localiser les éléments de contenus de manière plus globale et à l'aide de connaissances a priori exploitées dès les premières étapes du processus d'analyse. Elles sont souvent moins sensibles au bruit que les méthodes ascendantes. Néanmoins, elles sont difficilement applicables sur

les documents ayant une mise en page fluctuante ou non rudimentaire puisque les algorithmes exploités se basent sur des connaissances a priori [Hadjar02] (nombre de colonnes, critères d'arrêt du découpage,…). Ce type de techniques souffre alors du même problème de non généricité (que les méthodes ascendantes) et il est donc difficile d'utiliser ces dernières sans avoir à recoder complétement le module spécifiant le modèle du document ou les connaissances a priori à exploiter pour chaque lot d'images. L'analyse se base toujours sur un modèle qu'il est nécessaire de re-spécifier pour chaque classe de documents à traiter. La nécessité de re-concevoir ce modèle et le procédé d'instanciation associé pour chacune des classes de documents constitue alors une énorme difficulté qui empêche encore aujourd'hui la mise à disposition d'outils ou de méthodes génériques à même de traiter divers types d'images.

De nombreux systèmes exploitent des méthodes à base de grammaires et de règles pour encoder le modèle générique de document et son mode d'instanciation sur les images. Ce type de méthodes a été utilisé dans le cadre du projet européen Meta-e [Metae2013] pour produire le seul système commercial de reconnaissance de structure logique qui semble exister aujourd'hui. Malheureusement, l'approche descendante proposée n'est applicable que sur des structures rigides et n'a donc été appliquée qu'au traitement des livres allemand du XVIIIe. Dans le même ordre d'idée, la méthode DMOS permet de générer automatiquement un système de reconnaissance de documents structurés [Couasnon02]. Elle utilise une grammaire pour décrire la structure type du document à analyser couplée à un compilateur qui va utiliser cette grammaire pour générer un langage de description du document. Dans les deux cas, l'encodage des règles s'effectue manuellement en laboratoire par un expert.

Pour éliminer ou limiter cette phase manuelle, des tentatives de mise en place de méthodes de construction du modèle de documents par apprentissage ont été effectuées mais se sont avérées infructueuses à cause de la difficulté à disposer de données étiquetées en nombre suffisant.

Vers des systèmes interactifs

Afin de contourner les difficultés mentionnées, depuis peu, certains chercheurs s'intéressent à la mise en place de systèmes interactifs pour guider ou corriger la réponse du système au cours de la phase d'analyse.

Nous pouvons mentionner les travaux de [Roussel2001] qui propose un mécanisme d'interaction qui permet à l'utilisateur d'intervenir sur la structure produite progressivement par le système. Ceci permet d'influencer la réponse du système pour le reste de l'analyse de la page courante, en transformant manuellement certains éléments dans la structure produite. Il s'agit donc ici de corrections manuelles des erreurs commises par le système au plus tôt afin d'éviter qu'elles se propagent. Le modèle du document et l'enchainement des traitements prévus ne sont jamais remis en cause. La re-soumission de la même image provoquera les mêmes erreurs qu'il faudra à nouveau corriger manuellement.

Le système smartFIX [Klein2004] affecte un score de confiance aux éléments extraits automatiquement. Une validation manuelle est alors sollicitée pour les éléments dont le taux de confiance est trop faible. A nouveau, seule la possibilité de modification des résultats finaux est offerte. Il est impossible d'agir sur le processus d'analyse ou de guider l'interprétation durant ou même en fin de traitement.

Dans leurs travaux, [Chazalon2012] définissent deux modes d'interaction : le mode dirigé par les erreurs (qui correspond aux situations décrites précédemment permettant uniquement à l'utilisateur de corriger ou valider les résultats produits par la machine) et le mode spontané laissant l'utilisateur produire manuellement un résultat sans qu'il y ait forcément suspicion d'erreur. Dans ce deuxième mode, le système doit réagir à l'information externe qui lui est fournie pour améliorer sa réponse pour une image donnée. Ici encore, l'utilisateur fournit un résultat à la place du système mais ne modifie pas le processus d'analyse afin de l'améliorer pour les prochaines images a traiter.

Dans la plupart de ces systèmes, la capacité à détecter les erreurs est le critère essentiel influant sur l'interactivité. Nous pensons que cette stratégie est difficile à mettre en place et pensons qu'il serait plus profitable que l'interaction ait lieu en amont et/ou agisse directement sur le processus d'analyse pour l'améliorer plutôt que sur les résultats produits au cours du temps (correction d'erreurs ou production spontanée d'information).

En complément des approches préalablement décrites, nous proposons de permettre à l'utilisateur de définir l'ordre et les critères d'extraction des éléments pertinents (sans d'ailleurs forcément mettre en place un réel modèle des documents à traiter mais plutôt sur la base de ses connaissances sur les images à traiter) plutôt que d'attendre que le système génère des erreurs difficiles à corriger. Notons, que notre proposition reste compatible et complémentaire aux mécanismes d'interaction précédemment cités notamment le mécanisme de production spontanée d'information qui peut permettre de débloquer un processus d'analyse avant qu'il ne produise des erreurs.

La section suivante présente l'architecture requise pour mettre en place ce mécanisme permettant la production interactive de processus d'analyse de structures et d'extraction d'éléments de contenus. Chacun des processus produits étant ainsi construit « à façon » donc adapté et applicable sur divers types d'images de documents.

3. Modélisation des données

Lors de l'état de l'art, nous avons vu qu'un modèle à base d'arbre était classiquement employé pour représenter aussi bien les structures physiques que logiques des documents. De plus, nous verrons par la suite qu'une modélisation des données extraites des images sous forme d'arbre autorise :

- d'une part la spécification rapide du (ou des) contexte(s) sur lequel un traitement doit être appliqué au cours du processus d'analyse,

- et d'autre part l'exploitation d'un mécanisme d'appariement sous contraintes de sous-arbres utilisables pour détecter des contenus spécifiques.

Représentation des données sous forme d'arbre

Le cœur de l'architecture proposée repose sur une représentation du contenu des images à l'aide d'un arbre (*T*) d'éléments de contenu (EoC – Element of Content) pouvant évoluer au cours de l'analyse. Un *EoC* est une entité identifiée par le processus d'analyse.

Les nœuds de l'arbre représentent les éléments de contenu extraits de l'image durant l'analyse. Chaque nœud possède des attributs propres (décrits plus loin) permettant une spécification précise de ses caractéristiques (type, position, forme, …). Les arcs orientés définissent tous la relation « est composé de » permettant de mettre en place la relation « père-fils » dans l'arbre *T* résultant d'une analyse (figure 3). On pourra ainsi modéliser, par exemple, qu'un *EoC mot* (père) est composé de 5 *EoC caractères* (fils). Les arcs ne possèdent pas d'attribut.

Chaque nœud de l'arbre *T* possède les attributs suivants :

- un attribut *Label* symbolique permettant d'associer un type (pré-défini ou choisi par l'utilisateur) à chaque nœud (*Doc, CC, Word, Noise,* …)
- un attribut *Propriétés* correspondant à 2 (pour pouvoir spécifier des intervalles pour chaque caractéristique) vecteurs numériques de dimension *N*. Dans la version actuelle, les nœuds possèdent un ensemble de *N=5* caractéristiques (*X, Y, Xmap, Ymap, NbChild*) exprimée chacune par un intervalle [a;b] avec $a \leq b$

En début d'analyse, un arbre *T* initial comportant un nœud unique avec le *Label Doc* correspondant à toute l'image est créé (figure 3). L'utilisateur aura la charge de définir le processus d'analyse (scénario) chargé de compléter l'arbre *T* par extraction incrémentale d'éléments de contenu fils de divers types.

4. Opérateurs de transformation de l'arbre

Le système proposé se base sur un nombre très faible d'opérateurs élémentaires pour faire évoluer l'arbre courant *T*. Cependant, afin de simplifier la phase de mise en place interactive de scenarios d'analyse, trois niveaux d'opérateurs ont été mis à disposition des usagers dans la version actuelle d'Agora. Les opérateurs de niveau supérieurs pouvant faire appel à plusieurs opérateurs de niveau inférieur pour produire le résultat escompté. Un utilisateur expert serait donc à même de produire n'importe quel scénario en utilisant uniquement les opérateurs de niveau élémentaire sans avoir recours aux autres niveaux. Nous détaillerons donc, dans cet article, uniquement les opérateurs élémentaires qui constituent le cœur de l'architecture et passerons plus rapidement sur les opérateurs des niveaux supérieurs (bien que leur simplicité d'utilisation soit un point fondamental pour les utilisateurs – aspect IHM).

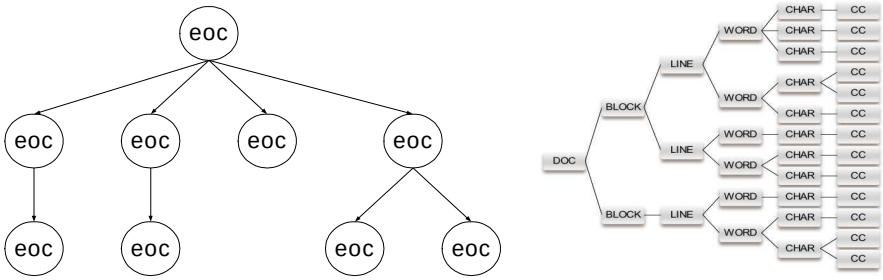


Figure 3 : Exemple d'arbre d'EoC générique et instancié/obtenu après une analyse

Opérateurs élémentaires

Les opérateurs élémentaires ont pour objectif de transformer l'arbre courant *T* représentant les éléments de contenu présents dans l'image en cours d'analyse.

Chacun des opérateurs élémentaires fonctionne suivant le modèle ci-dessous :

1. Recherche d'appariements sous-contraintes entre un *Pattern utilisateur* (arbre à 2 niveaux, avec de 0 à N fils) et des sous-arbres de *T*
2. Pour chaque appariement validé ➔ Réalisation de l'action associée à l'opérateur élémentaire sur le sous-arbre détecté (*Expand*, *Insert,* ou *Delete* – cf ci-dessous)

Afin d'alimenter l'arbre en éléments de contenu physiques extraits de l'image, le premier type d'opérateurs mis à disposition des utilisateurs est l'opérateur **Expand.**

Ce type d'opérateurs permet d'exploiter des algorithmes d'analyse d'images pour extraire des EoC physiques des images. Il peut s'agir par exemple d'extraire les composantes connexes après binarisation mais on peut imaginer offrir aux utilisateurs divers possibilités comme par exemple un opérateur *Expand* permettant l'extraction d'autres configurations spécifiques de pixels (segments, arcs, …).

Les opérateurs *Expand* nécessitent la spécification de 2 opérandes : le premier est le *Pattern utilisateur* définissant un motif à rechercher. Dans le cas d'*Expand*, ce pattern ne doit posséder aucun fils (il s'agit juste de spécifier les nœuds pères qui serviront de racine). Le second paramètre est le *Label* à attribuer aux EoC physiques créés sous ces racines. Par exemple, *Expand(Doc, «CC»)* permettra la détection de toutes les composantes connexes présentes dans une image afin d'alimenter l'arbre courant *T* comme illustré figure 4.

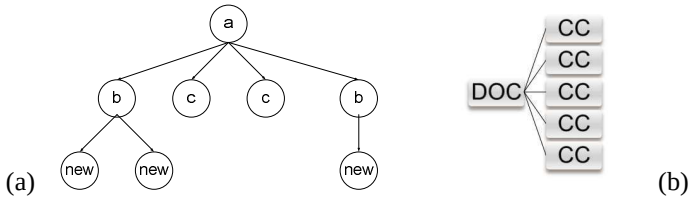


Figure 4 : Résultat de (a*) Expand(b, « New »)* et (b) *Expand(Doc, «CC»)*

Ensuite, l'opérateur le plus important de l'architecture proposée est probablement celui permettant l'insertion de nœud dans l'arbre conditionné par l'apparition de motifs spécifiques. Cet opérateur **Insert** prend 3 paramètres :

- un *Pattern* composé d'un père et de 1 à N fils
- le *Label* du nœud supplémentaire qui sera inséré entre le père et les fils identifiant ainsi l'apparition du motif recherché)
- les *Contraintes* devant être respectées par les *Propriétés* (caractéristiques numériques) des nœuds appairés pour que l'appariement soit complément validé.

Les paramètres *Pattern et Contraintes* sont définis par l'utilisateur de manière à spécifier un motif correspondant aux occurrences de sous-arbres à rechercher. Le *Pattern* ne peut comporter que deux niveaux : 1 nœud racine et de 1 à *N* nœuds fils. Les *Propriétés* et *Labels* de chacun des nœuds fils constituant le motif à rechercher sont définis par l'utilisateur afin de produire un motif très précis (pour par exemple détecter toutes les composantes connexes de petites tailles). Le mode de définition du motif à rechercher et des contraintes associées est décrit section 5. Lorsqu'un appariement est trouvé un nœud fils avec le *Label* spécifié est inséré dans l'arbre *T* tel qu'illustré sur la figure 5.

Afin de simplifier, l'arbre courant *T*, un opérateur **Delete** est mis à disposition de l'utilisateur pour supprimer de l'arbre *T* les EoC qui ne l'intéressent plus mais alourdissent les traitements (EoC de type bruit, zones textuelles si seules les lettrines importes, …). ou encore les EoC qui ont seulement servi à isoler temporairement un sous-ensemble d'EoC fils pour leur appliquer un traitement particulier.

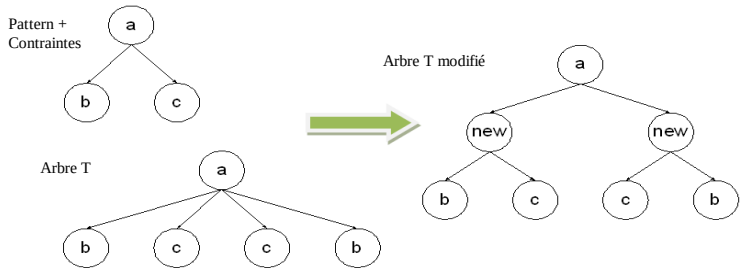


Figure 5 : Exemple simple d'utilisation de *Insert(New,Pattern, Contraintes).* L'insertion des nœuds *New* permet de marquer la présence du motif recherché dans *T*

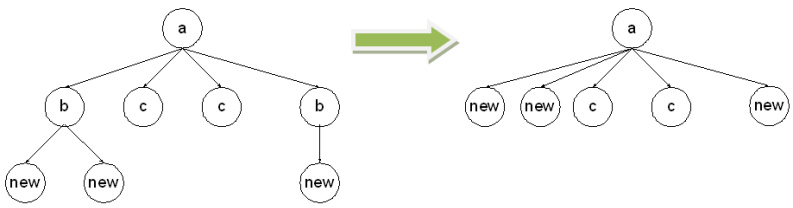


Figure 6 : Suppression d'EoC via l'opérateur *Delete(Pattern(a,b)).*

Les paramètres de cet opérateur sont le *Pattern* avec comme nœud père le *Label* des racines à explorer et comme nœud fils le *Label* des EoC devant être supprimés. Une variante permet de détruire toute la sous-branche concernée plutôt qu'uniquement le nœud fils (figure 6).

Lorsqu'un EoC est créé par insertion, il est nécessaire d'initialiser les valeurs de ses propriétés à partir des propriétés de ses nœuds fils. Un opérateur intitulé **SetFeature** a été développé pour réaliser ce travail lors de l'insertion de nœud mais aussi sur demande de l'utilisateur à n'importe quel instant du processus d'analyse de l'image (de la transformation de l'arbre courant). Pour chaque borne a ou b d'une caractéristique X d'un nœud, plusieurs modes de mises à jour sont possibles. Les modes de calcul suivants ont été implémentés (on note X.a et X.b les valeurs [a ;b] pour la caractéristique X) :

- No : caractéristiques X.a ou X.b inchangée
- ChildsMin : X.a et X.b père = minimums des caractéristiques X.a ou X.b des fils
- ChildsMax : X.a et X.b père =maximum des caractéristiques X.a ou X.b des fils
- ChildsAverage : X.a et X.b père =moyennes des caractéristiques X.a ou X.b fils
- ChildsAverageAll : X.a et X.b père =moyenne des caractéristiques X.a et X.b fils

Les paramètres de cet opérateur sont à nouveau le *Pattern* à rechercher par appariement. Les propriétés des nœuds fils validant le *Pattern* seront mises à jour.

Opérateurs de niveaux supérieurs (assistants)

Les opérateurs de niveaux supérieurs sont conçus par assemblage d'opérateurs de niveau élémentaire et seront utilisables à partir de l'interface d'Agora.

L'opérateur **FusionElementElement** a pour objectif de permettre la reconnaissance (par insertion d'un nœud supplémentaire dans *T*) d'une configuration particulière d'EoC, par exemple, pour rassembler 2 lettres formant un embryon de mot. L'opérateur **ClassifyByFeature** permet de reconnaître et d'extraire des EoC selon leur taille, position ou/et forme en exploitant leurs propriétés. L'opérateur **Intersection** simplifie l'action de recherche d'EoC ayant des chevauchements, utile lors de l'extraction de zones graphiques. Enfin, à un niveau encore plus élevé (niveau 3), un opérateur **TextSegmentation** beaucoup plus évolué a été conçu pour aider les utilisateurs non expert à réaliser l'extraction des blocs, lignes, mots et caractères en les assistant dans le choix des paramètres nécessaires à cette analyse sur les images à traiter (Voir figure 11).

Bien qu'il ne s'agisse pas d'un réel opérateur, il nous faut mentionner la fonctionnalité d'**exportation** des résultats qui permet de stocker l'arbre courant au format Alto en offrant aussi la possibilité d'extraction de toutes les imagettes étiquetées de l'ensemble des EoC reconnus grâce au scénario mis en place.

5. Principes d'appariement des sous-arbres

Le mode de recherche et comparaison des sous-arbres mis en place au sein de notre moteur d'analyse exploite une recherche d'appariement exact pour ce qui

concerne l'aspect structurel et tenant compte du *Label* des nœuds (structure d'arbre identique et correspondance exacte entre les étiquettes symboliques des nœuds) couplé à une méthode de validation de contraintes appliquée dans un deuxième temps sur les propriétés numériques de l'ensemble des nœuds mis en correspondance afin de valider complétement l'appariement trouvé vis-à-vis des spécifications de l'utilisateur (paramètres *Pattern + Contraintes*).

Recherche et analyse des appariements

Le principe de base de la mise en correspondance d'abord symbolique puis numérique dans un second temps est le suivant :

1. Recherche d'appariements exacts (aspect structurel et symbolique) entre le *Pattern utilisateur* (arbre à 2 niveaux, avec de 0 à N fils) et des sous-arbres de l'arbre courant *T*
2. Pour chaque appariement structurel trouvé, Analyse des *Contraintes* numériques imposées par l'utilisateur :
   - *a*. L'utilisateur a spécifié des contraintes vis-à-vis des valeurs de caractéristiques sur les nœuds fils du *Pattern utilisateur*
     - Recherche de la meilleure transformation linéaire contrainte $\mathbb{R}(A,B)$ permettant la mise en correspondance des nœuds du *Pattern utilisateur* avec les nœuds de *T* qui leur sont appairés.
     - Modification du *Pattern* selon la transformation trouvée $\mathbb{R}(A,B)$
     - Vérification des *Contraintes* pour chaque fils pris dans *T* et appairé avec un fils du *Pattern utilisateur* modifié
   - b. L'utilisateur a spécifié des contraintes entre les caractéristiques des fils sélectionnés dans *T* ➔ Vérification des *Contraintes* pour chaque couple de fils appairés pris dans *T*
   - *c*. L'utilisateur a spécifié des contraintes entre les caractéristiques du père et des fils sélectionnés dans *T* ➔ Vérification des *Contraintes* pour (Père, EoC fils) pour chaque fils appairé pris dans *T*
3. Validation définitive de l'appariement ➔ Retour booléen (Oui/Non)

La vérification des contraintes imposées par l'utilisateur entre les sous-arbres mis en correspondance s'effectue caractéristique par caractéristique, à partir de 4 règles de comparaison appliquées sur les intervalles [a ;b] de chaque caractéristique sélectionnée par l'utilisateur (figure 7). Ces 4 comparaisons sont réalisées sur les bornes des 2 intervalles à comparer ce qui revient à spécifier numériquement des valeurs de tolérance maximum correspondant à chacune des 13 relations spatiales possibles telle que définis dans [Allen1983].

Transformation linéaire

L'étape de transformation linéaire $\mathbb{R}(A,B)$ permet d'autoriser des déformations avant de tester les contraintes spécifiées par l'utilisateur.

• enableTranslation : à positionner à false pour rechercher des patterns concernant la position absolue des bornes des intervalles; à positionner à true pour rechercher des patterns concernant les longueurs des intervalles
• enableScaling : permet de s'affranchir de l'échelle lorsqu'il est positionné à true (à utiliser pour comparer des rapports entre intervalles par exemple)
• enableRotation : pas exploité pour l'instant

La recherche et l'application de la meilleure transformation $R(A,B)$ suit l'algorithme suivant :

Soit Y = l'ensemble des n couples de vecteurs associés aux nœuds pris dans *T* rassemblés dans une matrice de dimension (N,2.n) avec N=5 caractéristiques actuellement, n le nombre de fils appairés et (2 pour travailler avec des intervalles)
Soit X = l'ensemble des n couples de vecteurs associés aux nœuds du *Pattern utilisateur* recherché (matrice N x 2.n)

1. Recherche de la meilleure transformation linéaire $R(A,B)$ telle que $Y=A.X+B$
2. Application, sur le *Pattern*, de la transformation trouvée pour obtenir $\hat{Y}=A.X+B$
3. $\hat{Y}$ devient le nouveau *Pattern utilisateur* utilisé pour vérifier les contraintes

Lors de la définition des *Contraintes* utilisateurs, seules certaines des caractéristiques (choisies par l'utilisateur) doivent pouvoir être considérées (si par exemple seule la position en X importe à l'utilisateur). Cette fonction est rendue possible par la mise en place d'une matrice de projection permettant de spécifier les axes d'intérêt avant recherche de la meilleure transformation. Par exemple, lorsqu'on travaille sur les informations géométriques X et Y d'un EoC, on se moque du nombre de fils qu'il possède, on applique une matrice de projection ayant 0 dans les dimensions à ignorer.

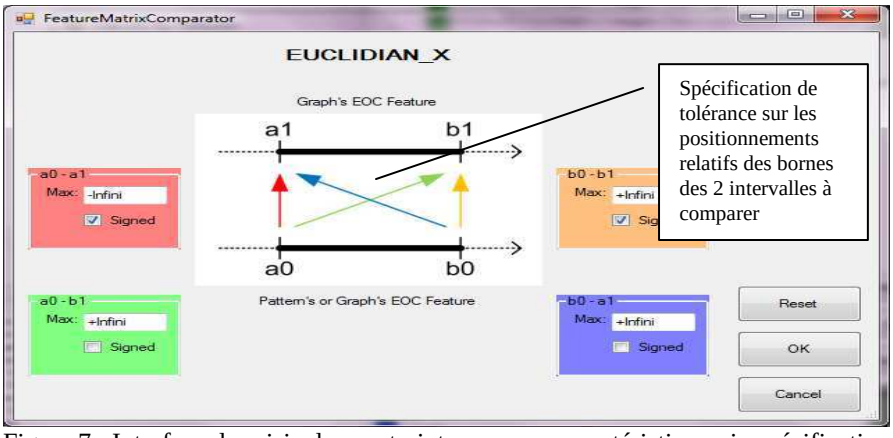


Figure 7 : Interface de saisie des contraintes sur une caractéristique via spécification de seuils de tolérance sur l'une des relations de Allen entre 2 intervalles

6. Mise en place interactive de processus d'analyse incrémentale

Méthodologie

Sur la base de la modélisation des données sous forme d'arbre et des opérateurs de transformation précédents, il est possible d'offrir à l'utilisateur les interfaces lui permettant de construire lui-même des chaînes de traitements (que nous nommerons scénarios d'analyse incrémentale) faisant évoluer progressivement l'arbre T représentant la structuration courante en fonction des caractéristiques des images à analyser et des objectifs visés. Le but de l'utilisateur devant être d'obtenir, à terme, un processus de construction de l'arbre le plus précis possible et applicable sur un lot d'images déterminé.

L'architecture proposée met à disposition de l'utilisateur des interfaces lui permettant de spécifier simplement les paramètres des opérateurs de transformation de l'arbre courant afin de produire des scénarios d'analyse de manière interactive. Les opérateurs mis à la disposition de l'utilisateur pour construire les processus d'analyse sont appliquées selon la stratégie définie par l'utilisateur qui peut :

- ne s'intéresser, dans un premier temps, qu'aux zones graphiques ou au contraire qu'aux zones textuelles,
- supprimer des zones d'un type particulier (par exemple les zones étiquetées Texte si l'utilisateur ne s'intéresse qu'aux lettrines)
- choisir d'étiqueter d'abord les zones facilement caractérisables à l'aide de règles d'extraction simples pour ensuite se baser sur un contexte plus riche pour extraire des objets moins facilement identifiables [Ramel2007].

Pour construire son scénario, l'utilisateur applique les opérateurs successifs (enregistrés dans le scénario) sur une image type qu'il a sélectionnée ; ces actions sont appliquées sur l'image et il visualise le résultat de chacune d'elles en temps réel et peut ainsi valider leur intérêt ou revoir leur paramétrage. Les actions sont traduites sous forme littérale dans une liste visible en permanence par l'utilisateur (voir figure 10). L'utilisateur peut annuler sa dernière opération pour modifier son scénario. Une fois le scénario considéré comme correct, celui-ci peut être enregistré dans un fichier pour archivage ou pour application sur un ensemble plus conséquent d'images (un ouvrage complet) lors d'un traitement par lot.

Exemples concrets d'utilisation

Notre logiciel a été fourni au CESR qui teste intensivement cette nouvelle version d'Agora. Une formation à l'utilisation du logiciel (création de scénarios) a été dispensée aux utilisateurs potentiels (chercheurs, historiens, informaticiens) afin qu'ils puissent produire des scénarios et par conséquent tester notre système d'analyse interactive d'images. Cette collaboration entre le CESR et le laboratoire a d'ores et déjà permis d'améliorer et de compléter les interfaces du logiciel pour mieux prendre en compte les besoins du terrain (mise en place d'opérateurs de niveau supérieur type « assistant »).

Plusieurs expérimentations ont été menées, soit par le personnel du CESR, soit au sein de notre laboratoire. La figure 10 illustre quelques vues de la mise en place d'un scenario simpliste dont la première étape extrait les composantes connexes des images (opérateur *Expand*), puis utilise l'opérateur de haut niveau *TextSegmentation* pour extraire les caractères avec et sans accent, mots, lignes et blocs de texte.

La nouvelle version d'Agora a également permis de traiter les 624 pages d'un ouvrage de Ronsard numérisé à 500dpi (figure 8) afin d'en extraire les zones de texte (paragraphes de proses, paragraphe de vers et lignes de texte dans ces paragraphes) pour générer le fichier Alto correspondant à chacune des pages. Le scénario utilisé comporte les grandes étapes décrites figure 9.

Figure 8 : Images de l'ouvrage de Ronsard

1. Extraction des composantes connexes CC par Expand
2. Détection des IMAGES (CC de taille supérieure à 100 pixels)
3. Détection et suppression des petites CC = NOISE afin de conserver uniquement de quoi construire grossièrement des lignes de texte et les images
4. CHAR = CC restantes
5. Construction grossière des LINE (= ensemble de CHAR alignés horizontalement)
6. Affinage de ces lignes : découpage d'une LINE en plusieurs LINE2 si CHAR très espacés
7. Elimination des LINE2 trop petites
8. Construction des BLOCK à partir des LINE2, et simplification des BLOCK qui s'intersectent
9. Construction d'un bloc englobant LAYOUT et différenciation PROSE/VERS selon l'alignement avec le bloc englobant
10. Exportation Alto

Figure 9 : Scénario Ronsard

6. Conclusion

Cet article décrit un nouveau moteur d'extraction d'éléments de contenu et d'analyse de structures de pages numérisées. Ce moteur a été implémenté au sein de la dernière version du logiciel Agora disponible au téléchargement sur le site du projet Paradiit [Paradiit2013].

Cette nouvelle méthode d'extraction de structures et d'éléments de contenu se base sur un mécanisme original d'appariement séquentiel de sous-arbres. Les sous-arbres permettent la description des structures à localiser dans l'état courant de l'interprétation du document représentée également par une structure arborescente. Un algorithme original de recherche d'appariements sous contraintes de sous-arbres pouvant être exploité de manière incrémental permet la mise en place interactive de scénarios d'analyse cherchant à répondre aux besoins spécifiques de chaque usager.

La seconde originalité de notre approche réside dans l'opportunité que nous offrons aux utilisateurs de pouvoir construire de manière interactive des scénarios d'analyse incrémentale d'images. Le but est, partant d'une segmentation initiale constituée d'éléments de contenu élémentaires (composantes connexes ou autres objets physiques pouvant être extraits directement des images), de laisser l'utilisateur concevoir sa chaine de traitement en faisant évoluer une représentation des images sous forme d'arbre d'éléments de contenu de manière progressive. L'objectif est

d'aboutir à un scénario d'analyse adapté fournissant une caractérisation la plus fine possible des contenus à reconnaitre en fonction des objectifs visés et du type d'images à analyser.

Le CESR teste actuellement ce nouveau prototype avec des objectifs variés allant de l'extraction d'éléments graphiques dans les ouvrages anciens à la création de modèles typographiques. Même si le système reste difficile à prendre en main sans une formation adéquate, ce travail a débouché sur la mise à disposition d'un unique outil (exécutable, bases de scénarios et code sources) pour divers types d'utilisateurs (traiteurs d'images, spécialistes de l'histoire du livre, informaticiens, etc.) utilisable dans de multiples contextes.

Bibliographie

[Allen1985] Allen J.F. and P. J. Hayes. A common-sense theory of time. In 9th International Joint Conference on Artificial Intelligence, pages 528-531, Los Angeles, 1985. IJCAI.

[Alaei2011] Alaei A, U. Pal, and P.Nagabhushan, "A new scheme for unconstrained handwritten text-line segmentation," Pattern Recognition, vol. 44, pp.917–928, 2011.

[Chazalon2012] Chazalon J., Coüasnon B., Lemaitre A., «Comment introduire simplement et uniformément deux modes d'interaction asynchrones complémentaires dans un système d'analyse de documents existant », CIFED, 2012.

[Coüasnon2002] Coüasnon B., J. Camillerapp, DMOS, une méthode générique de reconnaissance de documents : évaluation sur 60 000 formulaires du XIXe siècle, in Actes du Colloque International Francophone sur l'Écrit et le Document, Hammamet, 2002.

[Hadjar2002] Hadjar K, O. Hitz, L. Robadey, R. Ingold. Configuration REcognition Model for Complex Reverse Engineering Methods: 2(CREM). Proceedings of the 5th International Workshop on Document Analysis Systems. p469-479 2002

[Lebourgeois2003] Lebourgeois F, H Emptoz, E Trinh, Compression et accessibilité aux images de documents numérisés – Application au projet Debora. Document Numérique. Vol 7 n°3-4. p103-127. 2003

[OGorman1993] O'Gorman L. The Document Spectrum for Page Layout Analysis In IEEE Trans. On PAMI.15(11). p1162-1173. 1993

[Klein2004] Klein B., Dengel A., Fordan A., « smartFIX : An Adaptive System for Document Analysis and Understanding », in: Reading and Learning, vol. 2956 of LNCS, Springer, 2004.

[Metae2013] http://meta-e.aib.uni-linz.ac.at/

[Michard2000] Michard A., XML: langage et applications, Eyrolles 2000.

[Paradiit2013] https://sites.google.com/site/paradiitproject/

[Ramel2007] Jean-Yves Ramel, S. Leriche, M. L. Demonet, S. Busson: User-driven page layout analysis of historical printed books. IJDAR 9(2-4): 243-261 (2007)

[Roussel2001] Roussel N., Hitz O., Ingold R., « Web-based cooperative document understanding », ICDAR, p. 368-373, 2001.

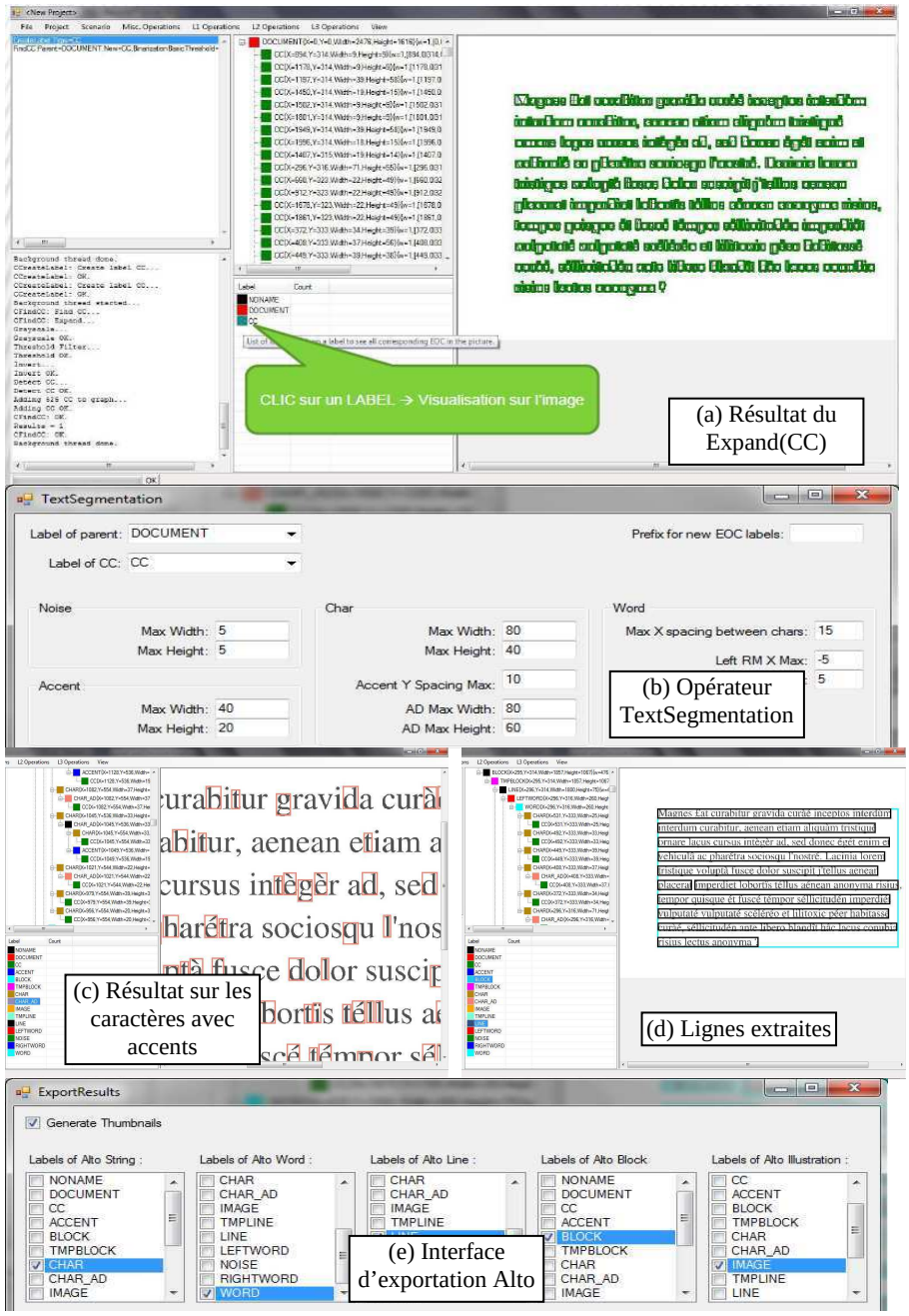


Figure 10 : Quelques vues de la mise en place d'un scénario simpliste exploitant l'assistant TextSegmentation