

Projet NTIC 2019

Détection graphique de lignes dans des manuscrits
anciens - Achilléides



Constantin Andreea & Zeller Quentin

18 mai 2019

Table des matières

1	Introduction	3
2	Matériel	7
3	Méthode	7
3.1	Généralité	7
3.2	Recommandation	8
3.3	Utilisation de l'application	9
3.4	Algorithmes	9
3.4.1	Détection des interlignes, commentaires	10
3.4.2	Nettoyage des erreurs de détection des initiales	12
3.4.3	Nettoyage des erreurs de détection des zones de texte	12
4	Discussion	13
4.1	Résultats et évaluation	13
4.2	Difficultés rencontrées	16
5	Conclusion	19

1 Introduction

Ce travail s'inscrit dans le contexte d'un projet de recherche sur des textes anciens, tels que l'Achilléide.¹ Notre but est d'améliorer des outils permettant l'affichage, la reconnaissance et l'analyse de ces textes. Plus concrètement, nous travaillons sur l'identification des lignes dans les manuscrits anciens scannés. Ceci afin de délivrer aux utilisateurs finaux des extraits de poèmes utiles pour leur analyse. Il y a plusieurs complications dans ce projet, notamment *les notations interlignes*, qui doivent pour la plus-part du temps être classées séparément des lignes du poème, et les *initiales* des lignes, qui des fois sont identifiées séparément du reste des lignes. Un cas d'utilisation est qu'une ligne de poème soit représentée par une seule ligne graphique textuelle à laquelle on peut attacher une transcription équivalente.

On travaille dans la logique globale de l'application illustrée dans la figure 1. La partie supérieure du schéma représente l'affichage des transcriptions pour le public, les chercheurs. Actuellement, elle se modélise sous la forme du site *achilleid* dont nous parlons plus tard. Pour l'identification des lignes de texte, le projet global utilise le logiciel *Transkribus*. C'est un logiciel open-source où il est possible de partager des documents/manuscrits avec d'autres utilisateurs. Le but est d'analyser et de reconnaître de manière semi-automatique des parties de textes anciens. Pour certains manuscrits, plusieurs versions de segmentation et transcriptions sont disponible sur les serveurs. Ces versions peuvent servir de modèle à notre application, qui aura besoin des document révisés comme référence (supervision).

La figure 2 montre le cas d'utilisation finale qui a pour but la création d'une édition critique de l'Achilléide accessible en ligne. Le site contient des images des manuscrits, des transcriptions et liens vers d'autres ressources liées à l'étude des classiques. Pour faciliter les transcriptions, il est important d'identifier de manière fiable des lignes de poème dans les images des manuscrits.

Dans la figure 3 on voit comment une page scanée d'un manuscrit est gérée par le logiciel *Transkribus*. Ce logiciel open-source permet la reconnaissance, la transcription et la recherche semi-automatique de documents historiques. Il se base sur la segmentation des images en blocs des texte et lignes, pour permettre d'attacher aux lignes du texte transcris. La figure 3 montre un exemple du problème d'écritures interlignes et en dehors du

1. "L'Achilléide (en latin *Achilleis*) est un poème épique latin inachevé de l'auteur Stace, qui devait présenter la vie du héros Achille depuis son enfance jusqu'à sa mort durant la guerre de Troie. Le premier livre et la moitié du second seulement (ce qui représente 1127 hexamètres dactyliques) furent rédigés avant la mort du poète. Subsistent le passage traitant de la jeunesse du héros, qu'il passa avec le centaure Chiron, et un épisode lors duquel sa mère Thétis le déguise en fille sur l'île de Scyros avant qu'il ne rejoigne l'expédition pour Troie." <https://fr.wikipedia.org/wiki/Achilléide>

bloc du poème. Il est nécessaire de classer ce type d'écritures séparément par rapport aux lignes de poème. L'image montre aussi la manière dont les lignes de texte sont identifiées graphiquement par des contours, sans distinction entre les types de lignes. Par contre, le cas problématique où une ligne de poème est identifiée par plusieurs lignes graphiques (par exemple l'initiale est séparée du reste de la ligne) n'apparaît pas dans cet exemple.

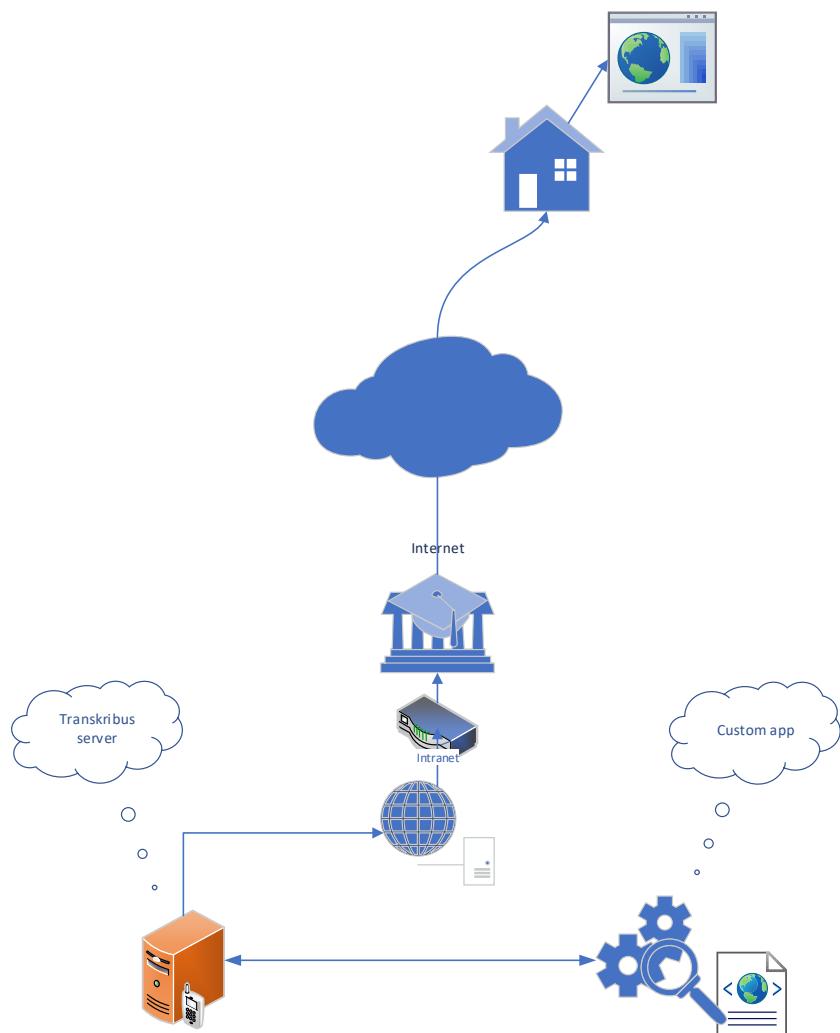


FIGURE 1 – .

ital Status: [The Poem](#) [The Manuscripts](#) [About](#)

Book Line

Click on an image to access the full page of the manuscript

Line 1.10 found in 7 documents

@54 Bern, Burgerbibliothek
156 – 88r

@70 Bruxelles, Bibliothèque Royale de Belgique
5337-5338 – 147r

@134 Eton, Eton College Library
150 – 18v

@369 München, Bayerische Staatsbibliothek
Clm 14557 (Ratisbonensis) – 77r

@467 Paris, Bibliothèque Nationale de France
lat. 8040 (Colbertinus) – 137r

Click on a verse to access all manuscripts containing this verse

1.1 magnanimum Aeaciden formidatamque Tonanti progeniem et patrio ueltiam succedere caelo diua refer quamquam acta uiri multum incita cantu Maeonio sed pura uacant nos ire per omnem

1.5 sic amor est heros uelis Scyroque latenter Dulichia proferre tuba nec in Hectore tracto sistere sed tota iuuenem deducere Troia tu modo si ueterem digno depleuimus haustu da fontes mihi Phoebe nouos ac fronde secunda

1.10 necete comas neque enim Aonium nemus aduena pulso nec mea nunc primis augescunt tempora uitte scit Dirceus ager meque inter prisa parentum nomina cumquaque suorum Amphione Thebae at tu quonge primum stupet Itala uirtus

1.15 Graiaque cui geminare florent utamque ducumque certatum laurus olim dolet altera uinc da ueniam ac trepidum patere hoc sudare parumper puluere te longo neccum fidente paratu molimur magnusque tibi praedulci Achilles

1.20 soluerat Oebalae classem de litore pastor Dardanus incutas blande populatus Amyclas plenaria materni referens praesagia somni culpatum relegabat iter qua condita ponto fluctibus iussus iam Nereis imperat Helle

1.25 cum Thetis Idaeos heu numquam uana parentum auguria expaut utrebo sub gurgite remos nec morte undosis turba comitate sororum prosulit thalamis feruent coeuntia Phrixii litora et angustum dominas non explicit aequor

1.30 illa ubi discussu primum sub aera ponto me petit haec mihi clavis ait funesta minatur agnoscit monitus et Protea uera locutum ecce nouam Priamo facibus de puppe leuat fert Bellona nurum video iam mille carnis

1.1 Le magnanime Achille, ce héros à qui le maître du tonnerre craignit de donner la vie, de peur de le voir un jour lui raser le trône du ciel, muse, c'est à toi de chanter. Ses exploits ont été illustrés par la lyre de Méomie ; mais le champ est vaste encore. Parcourir toute sa vie, celle est mon entreprise, l'aracher de sa retraite de Scyros, au bruit de la trompette d'Ulysse : laissez à Hector traîné dans la poussière : c'est loin de Troie que je veux montrer le jeune héros. Si jadis mes lèvres n'ont pas souillé les sources sacrées, permets-moi, ô Phébus ! d'y puiser encore, et ceins mon front d'une seconde couronne. Ce n'est point un hôte nouveau qui pénètre dans les bois d'Aonie ; ce n'est pas la première fois que les blanches bandelettes ornent ma chevelure. Les champs de Dirce me connaissent, parmi les noms de ses aieux. Thébes rédit mon nom et m'associe à son Amphion. Et toi qui contemple avec admiration l'élite et de l'Italie

1.15 et de la Grèce, toi pour qui les deux palmes du poète et du guerrier fleurissent à la fois, vainques tour à tour l'une par l'autre, pardonne-moi ; permets que quelque temps encore j'arrose de mes sueurs cette carrière. Par de longs et timides efforts je me prépare à chanter ta gloire, et le grand Achille sert de prélude.

1.20 Loin du rivage d'Ebaïe voguait le pasteur troyen, fier de la douce proie râvie à la confiante Amyclée ; déjà, accomplissant le présage du songe maternel, il traversait de nouveaux ces flots funestes que, du fond de la mer où elle a été plongée, Helle, nouvelle Néréide, gouverne à regret,

1.25 lorsque Thétis (hélas ! les pressentiments d'une mère ne trompent jamais), du fond de l'abîme azuré, tremble au bruit retentissant des rames. Soudain, suivie de la foule de ses soeurs, elle s'élançe de sa couche. Les rivages resserrés de Phryxus bouillonnent, et la mer est à peine assez large pour le cortège divin.

1.30 A peine Thétis eut-elle écarté les flots et touché les airs : « C'est contre moi qu'est dirigée cette flotte, » s'écria-t-elle, « c'est moi qu'elle menace. Je reconnais des prédictions funestes, et Protée m'a dit vrai. Voici qu'à la lueur des flambeaux élevés sur la poupe, à Priam Bellone conduit une fille nouvelle. Déjà mille vaisseaux couvrent

Latin text : <http://latin.packhun.org/loc/1020/3/0#0> slightly modified and without punctuation – Translation : French translation by M. Nisard, Stace, Martial, Manilius... Paris 1865, slightly modified

FIGURE 2 – Le site web *achilleid.unige.ch*, qui offre une édition digitale en accès libre à l'Achilléide.

Transkribus v1.6.0 (28.02.2019, 14:59), Loaded doc: 6_ANTWERPEN_duplicated ID: 133609, Page 1, file: 6_ANTWERPEN_01.tif [Image Meta Info: (Resolution300.0, w:h: 5476 * 3969)] current line: w:h: 882 * 172]

Server Overview Layout Metadata Tools

Logout andrea.pilois@gmail.com

Document Manager User Manager

Versions Jobs

Recent Documents... User activity

Collections: achilleid_ntic (3587, Owner)

1-3 / 3 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 800 Filter

1.1 magnanimum Aeaciden formidatamque Tonanti

1.2 progeniem et patrio ueltiam succedere caelo

1.3 diua refer quamquam acta uiri multum incita cantu

1.4 Maeonio sed pura uacant nos ire per omnem

1.5 sic amor est heros uelis Scyroque latenter

1.6 Dulichia proferre tuba nec in Hectore tracto

1.7 sistere sed tota iuuenem deducere Troia

1.8 tu modo si ueterem digno depleuimus haustu

1.9 da fontes mihi Phoebe nouos ac fronde secunda

1.10 necete comas neque enim Aonium nemus aduena pulso

FIGURE 3 – Le logiciel *Transkribus*, qui offre une plateforme avancée pour la transcription des manuscrits.

Nous nous trouverons donc dans la partie backend de l'application, voir *custom app* sur le schéma représenté dans la figure 1. Nous récupérerons les documents pré-analysés sur *Transkribus* et y ajouterons un post-traitement pour réparer les erreurs de segmentation connues de l'application. Nous ferons une preuve de concept qui pourra être directement intégrée dans le logiciel *Transkribus*.

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <PcGts xmlns="http://schema.primaresearch.org/PAGE/gts/pagecontent/2013-07-15" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   <Metadata>
4     <Creator>prov=University of Rostock/Institute of Mathematics/CITIlab/Tobias Gruening/tobias.gruening@uni-rostock.de:name=/net_tf/LA
5 TRP</Creator>
6   <Created>2018-01-12T12:46:04.329+01:00</Created>
7   <LastChange>2019-01-11T15:48:38.507+01:00</LastChange>
8 </Metadata>
9 <Page imageFilename="24_BERGAMO_24.tif" imageWidth="2429" imageHeight="3163">
10  <ReadingOrder>
11    <OrderedGroup id="ro_1547218118546" caption="Regions reading order">
12      <RegionRefIndexed index="0" regionRef="r1"/>
13    </OrderedGroup>
14  </ReadingOrder>
15  <TextRegion orientation="0.0" id="r1" custom="readingOrder {index:0;}">
16    <Coords points="902,73 902,2299 2187,2299 2187,73"/>
17    <TextLine id="r1l2" custom="readingOrder {index:0;}">
18      <Coords points="950,291 1003,294 1057,297 1110,300 1164,301 1217,304 1271,306 1325,306 1378,307 1432,309 1485,310 1539,311 1592,312
19      <Baseline points="950,272 1003,275 1057,278 1110,281 1164,282 1217,285 1271,287 1325,287 1378,288 1432,290 1485,291 1539,292 1592,293
20      <TextEquiv>
21        <Unicode>ferre et Amazonio conubia pellere ritu</Unicode>
22      </TextEquiv>
23    </TextLine>
24    <TextLine id="r1l3" custom="readingOrder {index:1;}">
25      <Coords points="950,373 998,376 1047,381 1095,384 1144,385 1193,387 1241,388 1290,390 1339,390 1387,390 1436,390 1484,391 1533,391 1567,398 1623,481
26      <Baseline points="950,354 998,357 1047,362 1095,365 1144,366 1193,368 1241,369 1290,371 1339,371 1387,371 1436,371 1484,372 1533,372
27      <TextEquiv>
28        <Unicode>sed mihi curarum satis est pro stirpe uirili</Unicode>
29      </TextEquiv>
30    </TextLine>
31    <TextLine id="r1l4" custom="readingOrder {index:2;}">
32      <Coords points="947,466 1003,467 1059,469 1116,470 1172,470 1228,472 1285,473 1341,474 1397,475 1454,476 1510,478 1567,478 1623,481
33      <Baseline points="947,446 1003,447 1059,449 1116,450 1172,450 1228,452 1285,453 1341,454 1397,455 1454,456 1510,458 1567,458 1623,458
34      <TextEquiv>
35        <Unicode>haec calathos et sacra ferat tu frange regendo</Unicode>
36      </TextEquiv>
37    </TextLine>
38    <TextLine id="r1l5" custom="readingOrder {index:3;}">
39      <Coords points="939,554 996,554 1053,556 1110,556 1167,557 1224,557 1281,559 1339,560 1396,562 1453,563 1510,565 1567,566 1624,568 1624,548
40      <Baseline points="939,534 996,534 1053,536 1110,536 1167,537 1224,537 1281,539 1339,540 1396,542 1453,543 1510,545 1567,546 1624,548
41      <TextEquiv>
42        <Unicode>indocilem sexuque tene dum nubilis aetas</Unicode>
43      </TextEquiv>
44    </TextLine>
45    <TextLine id="r1l6" custom="readingOrder {index:4;}">
```

FIGURE 4 – Exemple de fichier xml qui contient les informations sur la segmentation des images en blocs et lignes et la transcription équivalente.

Pour un travail futur, nous pensons que regrouper ces deux méthodes permettra d'apporter une solution robuste et automatique, épargnant ainsi du temps de travail laborieux aux personnes qui font la correction et la classification des détourage de ligne de texte.

2 Matériel

Cet outil est testé sur l'environnement suivant :

1. **OS** : Mac Os X, Version Mojave & Windows 10, version 1903.
2. **Langage** : Python 3.7 et Python 3.6 (incompatible python 2.X)

Les outils nécessaires sont :

1. Github et l'outil Git : pour le téléchargement des codes source
2. Python 3.7 ainsi que les sources suivante :
 - (a) TranskribusPyClient : une librairie permettant les requêtes avec Transkribus et développée directement par la communauté Transkribus.
 - (b) XML : proposé par Python directement. Attention, cet outil n'est pas conseillé dans un environnement sensible au niveau sécurité.²
 - (c) Numpy : une librairie mathématique.
 - (d) Re : pour les expression régulières.
 - (e) Matplotlib : pour l'affichage des graphiques / debugging.
 - (f) Et d'autres librairies incluses de base dans les installations Python conventionnelles.

Le programme a été testé sur des ordinateurs portables conventionnels. Le temps d'exécution est de l'ordre de 15 secondes pour un document de 50 pages. La majorité du temps est dépensé par les échanges avec le serveur Transkribus.

3 Méthode

3.1 Généralité

Nous avons désigné l'application uniquement sur les fichiers *XMLs* générés pour le logiciel *Transkribus*. Elle se base sur les erreurs produites par l'application et repose sur la connaissance de la structure sous-jacente des documents analysés.

Nous partons du principe que les documents sont constitués que de quelques zones de textes bien structurées, ce qui n'est de loin pas le cas de tous les manuscrits. Par conséquent, nous enlevons toute identification de texte qui n'est pas cohérente avec les zones de texte structuré.³ Nous admettons aussi que les espacements de lignes sont formatés

2. "Avertissement : Les modules XML ne sont pas protégés contre les données mal construites ou malicieuses. Si vous devez parcourir des données douteuses non authentifiées voir les sections *Vulnérabilités XML* et les paquets *defusedxml* et *defusedexpat*." docs.python.org/fr/3/library/xml.html

3. On couvre en détail dans la section 3.4 les critères que nous avons définis.

proprement. Toutes les lignes ne respectant pas ce format sont donc annotées comme interligne, qui est le but du projet.

Nous résolvons aussi quelques erreurs de détection très récurrentes, notamment les initiales des lignes qui ne sont majoritairement pas reconnues par le programme lorsque celle-ci sont majuscules et avec un espacement plus important par rapport au reste de la phrase. Ainsi, nous proposons une solution pour les inclure et dont nous parlerons dans la section suivante.

3.2 Recommandation

Nous recommandons d'utiliser la fonction de détection de zones de textes qui est en réalité un réseau de neurone prénommée "*preset*". Dans le cas où les zones de textes ne seraient pas bien définies, il est possible de les définir manuellement en amont de cette fonction ce qui permet une meilleure détection et avant tout une bonne attribution des lignes de textes aux zones de textes. Notre programme en aura besoin pour son bon fonctionnement de cette information. La figure 5 nous montre le potentiel problème de détection de deux colonnes en une seule colonne.

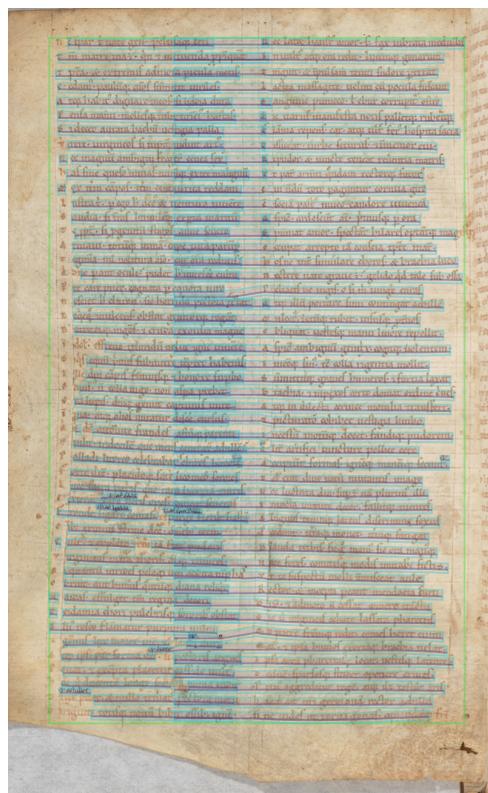


FIGURE 5 – Mauvaise détection des zones de textes dans *Transkribus*.

3.3 Utilisation de l'application

L'application se lance par ligne de commande. Elle prend un certains nombre d'arguments et dispose d'un invité d'aide comme on peut le voir sur la figure 6

```
MacBook-de-Quentin:src quentinzeller$ ./runTSCleaner.py -h
usage: runTSCleaner.py [-h] [--user USER] [--password PASSWORD]
                      [--colId COLID] [--docId DOCID]
                      [--interline-annotation INTERLINE_ANNOTATION]

optional arguments:
  -h, --help            show this help message and exit
  --user USER, -u USER  Your email account for Transkribus
  --password PASSWORD, -p PASSWORD
                        Your password for Transkribus
  --colId COLID, -c COLID
                        The collection ID you wan to modify
  --docId DOCID, -d DOCID
                        The document id you want to modify
  --interline-annotation INTERLINE_ANNOTATION, -i INTERLINE_ANNOTATION
                        To change the default tag for interline notation,
                        "footnote"
```

FIGURE 6 – Arguments d'entrée du script principal.

Ainsi, un exemple de commande serait :

```
#!/bin/bash
./runTSCleaner.py -u bob -p "mot_de_passe" \
-c 10 -d 10 -i "interlignes"
```

Attention, sur les distribution Unix, ne pas oublier de rendre le script exécutable par la commande :

```
chmod u+x runTSCleaner.py
```

L'environnement hôte doit avoir Python 3.X et les dépendances installées comme référencé dans la section Matériel. Lors de l'exécution du programme, certains avertissements à propos des requêtes peuvent apparaître. Ceci est normal et provient du fait que la librairie Transkribus-Python ne vérifie pas les certificats des requêtes HTTPS. Pour supprimer ces avertissements il faut installer les certificats des serveurs Transkribus. Dont le module *certifi*⁴

3.4 Algorithmes

L'algorithmique de l'application s'occupe de deux principales tâches, énoncées dans le cahier des charges : la détection des commentaires / écriture d'interligne et la liaison

4. user-guide.html

des initiales des lignes au reste des lignes. En plus de ceci, nous épurons les fautes de détection de zones de texte et nous adressons en partie les fautes de non-détection d'initiales.

Pour chaque page de manuscrit on commence par identifier les régions de texte correctes, que l'on garde pour faire le traitement des lignes. On applique les algorithmes suivants pour chaque région de texte séparément.

3.4.1 Détection des interlignes, commentaires

Pour ce faire nous utilisons les patterns récurrents dans les textes. Par chance, la majorité des manuscrits sont écrits d'une manière qui respecte une symétrie par rapport aux zones de textes et aux interlignes. Bien entendu, les textes étant écrits à la main, cela arrive souvent d'avoir des annotations interlignes ou même de marge. Dans certains manuscrits il est possible de voir plus de trois annotations entre deux lignes de texte correct.

Aux niveau des étapes algorithmique nous procédons comme suit :

1. Détection des lignes relativement aux zones de texte.
2. Calcul de la longueur des lignes.
3. Calcul de la distance des lignes par rapport à la marge gauche des zones de texte.
4. Détection des lignes qui sont en même temps trop courtes et trop éloignées de la marge gauche. On signale ces lignes comme des commentaires interlignes.

Pour décider si les lignes sont trop courtes ou trop éloignées de la marge gauche, on applique des seuils empiriques. Ces seuils sont estimés par rapport aux distributions des longueurs et distances. Dans les Figures 7 et 8 on peut voir que les lignes se séparent en distributions assez bien discriminantes. On combine alors l'information obtenue de ces deux statistiques, pour séparer les interlignes et les problèmes de détections des initiales.

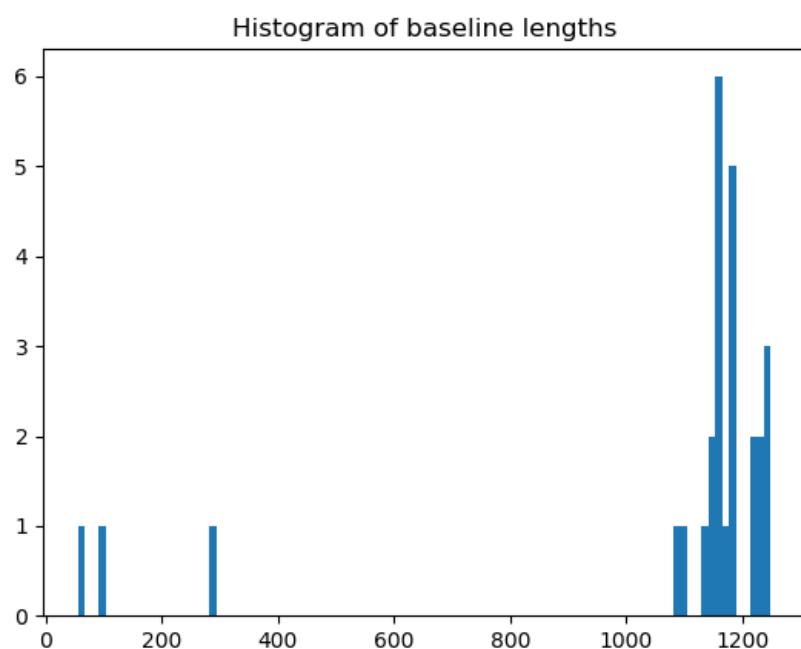


FIGURE 7 – .



FIGURE 8 – .

3.4.2 Nettoyage des erreurs de détection des initiales

Beaucoup d'erreurs récurrentes apparaissent dans le moteur *Transkribus*. La principale critique, déjà énoncée, était la détection séparée erronée des initiales des lignes. En plus, on approche aussi les cas où les initiales n'ont pas du tout été détectés.

Nous résolvons ces problèmes par une détection des lignes trop courtes et proches de la partie gauche des zones de textes. Si ces petites lignes sont en amont de plus grandes lignes de textes, elles sont fusionnées aux plus grandes lignes.

Aux niveau des étapes algorithmique nous procédonns comme suit :

1. Détection des lignes courtes et proches de la marge gauche. On signale ces lignes comme des initiales pour les fusionner aux lignes longues correspondantes.
2. Identification des lignes longues de texte se trouvant juste après les initiales (en termes de distance horizontale). Si une initiale se trouve entre deux lignes longues, on sélectionne la ligne longe dont la distance verticale par rapport à l'initiale est la plus petite.
3. Fusion des initiales avec les lignes longues associées.

Toujours dans le contexte de détection d'initiales, on traite le problème où les initiales des lignes n'ont pas été détectées. Pour ce cas, on doit recalculer la longueur des lignes après la fusion des initiales. On allonge le début des lignes longues de texte jusqu'à une marge verticale estimée en fonction d'où commence la majorité des lignes longues.

3.4.3 Nettoyage des erreurs de détection des zones de texte

Une autre fonction de nettoyage permet d'enlever les zones de textes trop petites ou insensées. Par exemple, il est possible que des dessins ou des taches soient détectées comme du texte et situées dans des zones très éloignées du texte principal. Ces zones de texte ne respectent en général aucune structure. On applique les étapes algorithmiques suivantes :

1. Reconnaissance des zones de textes et de leurs tailles.
2. Calcul de la largeur et de hauteur des zones de texte. Choisir comme référence les plus grandes de chaque.
3. Si une zone de texte est trop petite en largeur et hauteur par rapport aux références, on l'enlève.

4 Discussion

4.1 Résultats et évaluation

Notre solution est d'autant plus performante que les pages à analyser sont bien structurées, ce qui est normal étant donnée qu'on se base sur des filtres géométriques. Pour certaines pages on réussit à faire un traitement parfait, alors que dans d'autres cas les règles implémentées ne sont pas aussi fiables.

Pour attester de l'efficacité et de l'utilité des solutions qu'on propose, nous avons procédé à quelques évaluations de notre outil sur une sélection de trois manuscrits avec les particularités suivantes :

1. **369 Paris**, deux colonnes de texte, problèmes : initiales manquantes.
2. **24 Bergamo**, une colonnes de texte, problèmes : initiales manquantes.
3. **36 Berlin**, une colonnes de texte, problèmes : beaucoup d'interlignes et initiales manquantes.

La figure 9 montre un exemple de page de manuscrit (*Bergamo*) où notre logiciel a un comportement optimal. Nous avons à gauche la version du document avant avoir été traitée avec notre logiciel et à droite la version après le traitement. Même si c'est pas visible dans l'image, notre algorithme arrive à bien identifier aussi toutes les notations interlignes. On peut remarquer les traitements suivants :

1. Les petites régions de texte identifiées en vert ont été enlevées
2. Les lignes de texte avec l'initiale séparée ont été fusionnée aux initiales
3. Les lignes de texte sans initiale ont été élargies pour inclure leur initiales.

Nous avons calculé pour les pages 8 à 13, (pages contenant de l'information pertinentes), 42 erreurs d'initiales, 23 interlignes ou lignes non pertinentes, 10 TextBoxes erronées. Nous avons en sortie de notre algorithme 2 initiales erronées, et 3 lignes principales marquées comme interlignes. En résumé, ceci nous fait :

- 95% de réussite pour les initiales.
- 87% de réussite dans l'annotation d'interlignes.
- 100% de réussite dans la détection de TextBoxes non pertinentes.

Nous noterons que les erreurs sont apparues à cause d'une lettrine de début de chapitre, sans quoi ces 6 pages auraient été traitée correctement.

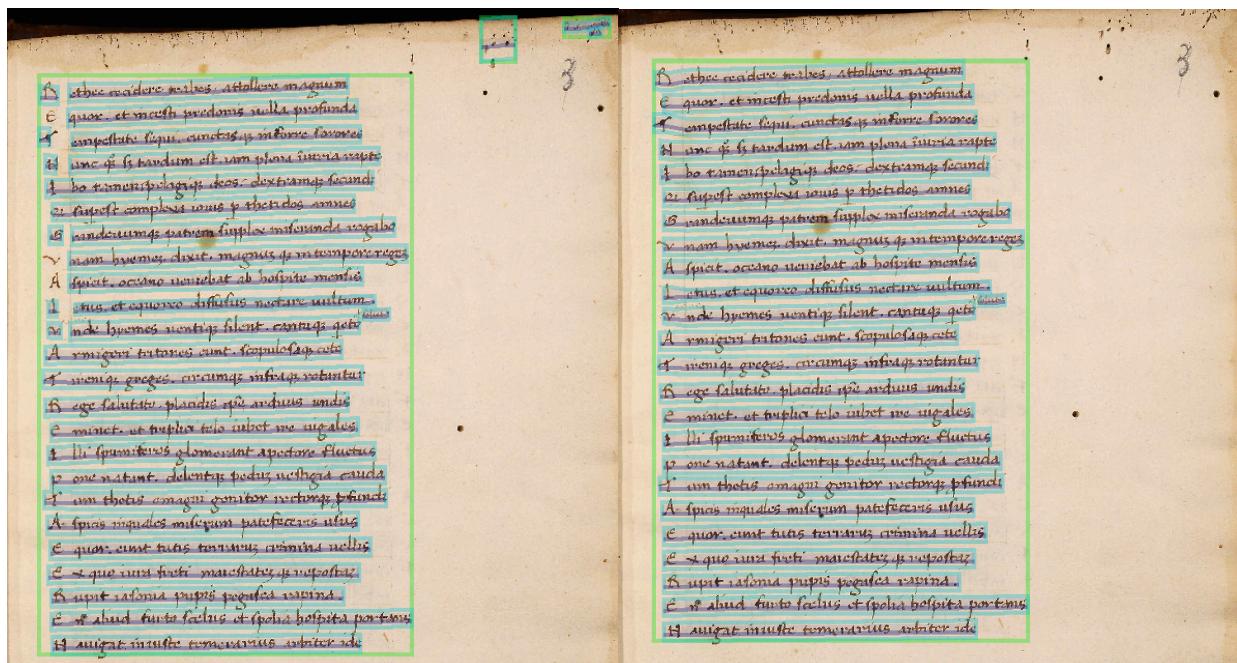


FIGURE 9 – Exemple de détection des commentaires interlignes avec notre logiciel.

La figure 10 montre la détection des commentaires interlignes sur un manuscrit difficile à traiter (*Berlin*). Dans ce cas, notre solution réussit à trouver 68% des commentaires interlignes et on peut voir que ces commentaires sont indiqués plutôt dans la partie droite du document. Cela s'explique par le fait qu'on impose des distances minimales fixes pour filtrer les écritures interlignes, et la performance de notre solution dépend des seuils choisis.

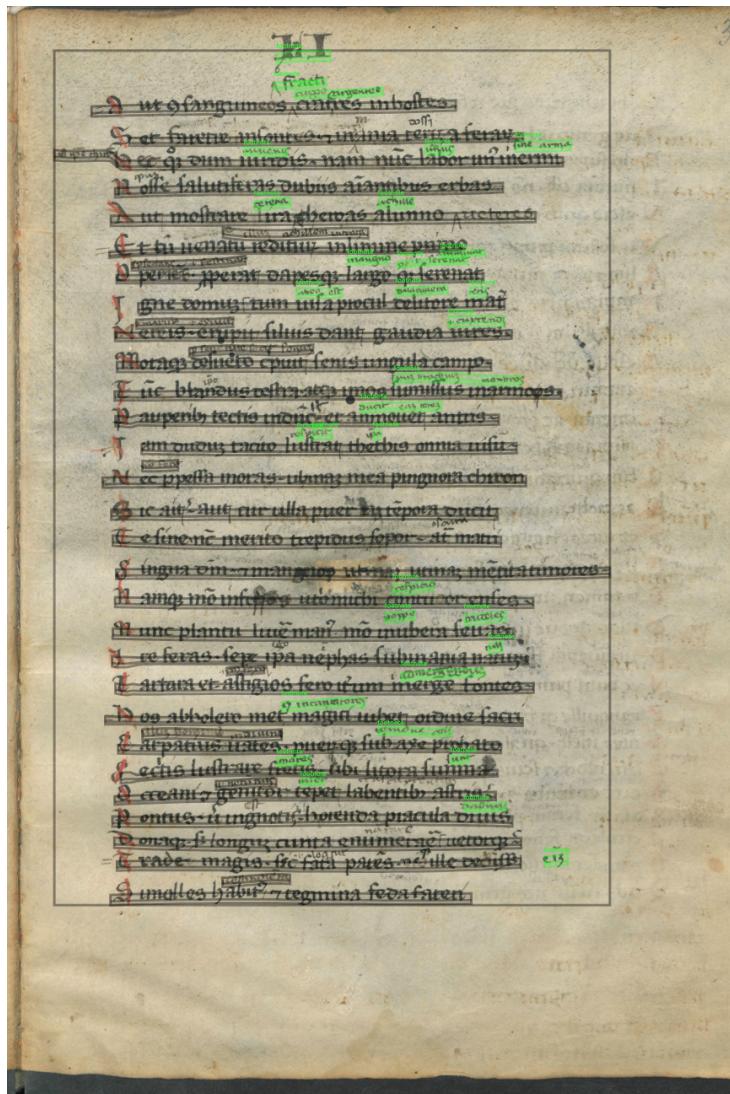


FIGURE 10 – Problèmes avec les commentaires interlignes.

La figure 11 montre le comportement de notre logiciel dans le cas de plusieurs régions de texte pertinentes. Dans ce cas, notre solution a un très bon comportement sur la page de gauche mais on peut voir sur la page de droite que, si la proportion des initiales non-identifiée est trop importante, nos algorithmes ne vont pas allonger les lignes pour inclure les initiales manquantes. Une autre limitation se relève au niveau des lignes de texte très courtes, qui risquent d'être vues comme des commentaires. Notre solution traite chaque région de texte séparément, mais on peut envisager une amélioration des algorithmes si on calcule les seuils nécessaires au filtrages des lignes une seule fois sur toutes les lignes de la page.



FIGURE 11 – Exemples de pages de manuscrit traitées avec notre logiciel.

Notre algorithme à été lancé sur les fichiers générés par le réseau de neurones dénommé *preset* de la méthode *CITLab advanced*. Un autre réseau de neurones *Konzilprotokolle Greifswald* génère beaucoup moins de problèmes d’initiales non détectées mais cependant il dispose d’énormément d’autres erreurs de détection. Il n’est vraisemblablement pas utilisé dans toute les collections que l’on a à disposition. De ce fait il n’est pas pertinent.

4.2 Difficultés rencontrées

Pour ce travail il a fallu trouver un algorithme qui puisse être compatible avec le maximum de documents et avec le moins d’interaction utilisateur possible. La plus grande difficulté était donc de trouver un algorithme simple pouvant s’adapter aux différents cas et aider la laborieuse tache de nettoyages des documents. En effet, la totalité des documents possèdent une structure différente. Par exemple, certains documents sont très bien formatés et respectent des règles géométriques strictes, alors que d’autres sont écrits de manière très informelle avec des début de phrases qui varient d’une ligne à l’autre. Ces cas sont exposés dans la figure 12.



FIGURE 12 – Exemple d'image sans structure.

Les problèmes réglés dans les algorithmes que nous avons générés tendent donc à résoudre les tâches les plus gourmandes en temps et facilement automatisables. L'exemple représentatif est la concaténation des initiales des lignes que nous pouvons voir sur la figure suivante.

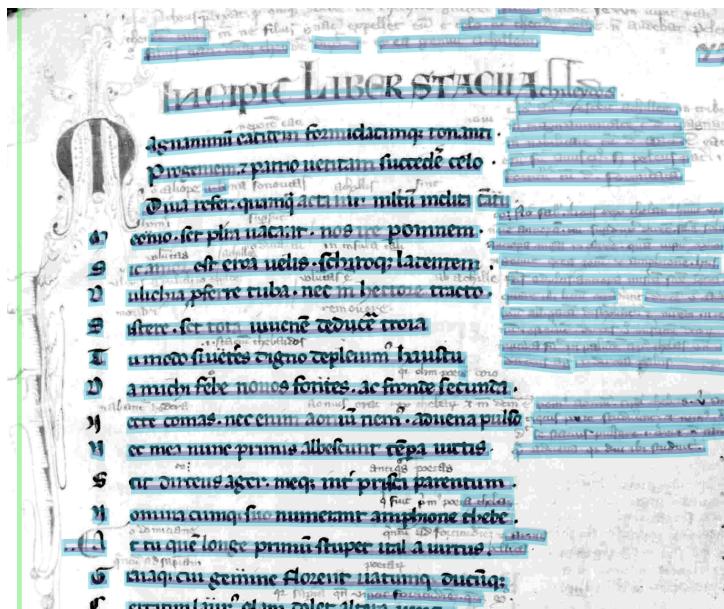


FIGURE 13 – Problèmes avec les initiales.

C'est une tâches laborieuse à régler à la main mais qui peut, dans beaucoup de cas être corrigée de manière automatique. Dans les Achilléides, beaucoup de documents présentent

cette particularité. C'était donc une tâche principale dans le cahier des charges.

Hormis ces éléments, nous avons du comprendre le fonctionnement des fichiers *XMLs* générés par *Transkribus* (cf. la figure 4). Ils sont bien structurés mais quelques détails peu clairs de prime abord sont importants. Toutes les lignes sont représentées par deux types de chaînes de coordonnées dont la variabilité ainsi que l'espacement change d'un document à l'autre et dépend de la résolution des images sous-jacentes. Les deux types de chaînes sont pour souligner les lignes (*baseline*) et pour les contourer (*linebox*). C'est pour ceci que dans la première phase de développement de notre application nous avons considéré dans une première étape uniquement les *baselines* du texte, les *lineboxes* étant plus compliquée et moins prévisibles.

Aussi, le déverminage est très compliqué, dans le sens qu'il est très dur de remarquer dans des chaînes de nombre très longues où les erreurs se situent. Sans compter que dans le *XML*, il y a une multitude de balise tout aussi semblable les unes que les autres pour les humains. La solution la plus complètes et facile s'avérait ainsi être la visualisation dans *Transkribus* même, avec l'interface graphique. Chose qui prend énormément de temps à procéder.

Finalement, nous avons constaté que l'ordre dans lequel on applique les filtres décrits dans les algorithmes est important pour le résultat final. Par exemple, fusionner les initiales au lignes de texte change la longueur de ces dernières et on doit recalculer la longueur et position relative des lignes avant de traiter les cas où les initiales ne sont pas du tout identifiées.

5 Conclusion

Le cadre de ce projet est lié à l'identification fiable des lignes dans les manuscrits anciens scannés. Le but final est de délivrer aux utilisateurs finaux des extraits de poèmes utiles pour leur analyse et transcriptions. Nous récupérerons les documents pré-analysés sur *Transkribus* et y ajouterons un post-traitement pour réparer les erreurs de segmentation connues de l'application. Un cas d'utilisation est qu'une ligne de poème soit représentée par une seule ligne graphique textuelle à laquelle on peut attachée une transcription équivalente.

Il y a plusieurs tâches que l'on résout dans ce projet, les plus importantes étant le marquage des *notations interlignes*, qui doivent pour la plus-part du temps être classées séparément des lignes du poème, et la fusion des *initiales* des lignes, qui des fois sont identifiées séparément du reste des lignes de texte. On adresse en plus l'épuration des fautes de détection de zones de texte et on offre une solution partielle pour les fautes de non-détection des initiales.

La méthode qu'on a choisi d'implémenter se base uniquement sur les fichiers *XML* fourni par l'application *Transkribus*. Dans notre cas, ces fichiers contiennent de l'information spécifique à la segmentation d'une image de manuscrit en bloc de texte et lignes. Notre méthode n'utilise pas de traitement d'image mais se contente d'analyser directement les coordonnées 2D des zones et lignes de texte pour y trouver des erreurs récurrentes.

En général, on crée des filtres pour les blocs et les lignes de texte en fonction de leur hauteur, longueur et position relative. On signale comme des commentaires interlignes toutes les lignes qui sont en même temps trop courtes et trop éloignées de la marge gauche du bloc de texte qui les contient. Pour fusionner les initiales, on commence par détecter les lignes courtes et proches de la marge gauche. Pour ces lignes on identifie les lignes longues correspondantes se trouvant juste après les initiales et qui ont la distance verticale par rapport à l'initiale la plus petite. Finalement on fusionne les initiales avec les lignes longues associées.

Notre solution est d'autant plus performante que les pages à analyser sont bien structurées, ce qui est normal étant donnée qu'on se base sur des filtres géométriques. Pour certaines pages on réussit à faire un traitement parfait, alors que dans d'autres cas les règles implémentées ne sont pas aussi fiables. Cela s'explique par le fait qu'on impose des seuils fixes pour filtrer les écritures interlignes et les initiales, et la performance de notre solution dépend de ce seuils. Pour l'avenir, on peut explorer une manière plus robuste pour choisir ces seuils, tout en utilisant les mêmes algorithmes qu'on a déjà développé.