Reconnaissance de structures logiques par un réseau de neurones transparent

Yves Rangoni, Abdel Belaïd

▶ **To cite this version:**

Yves Rangoni, Abdel Belaïd. Reconnaissance de structures logiques par un réseau de neurones transparent. Colloque International Francophone sur l'Ecrit et le Document - CIFED 06, Sep 2006, Fribourg, Suisse. pp.1 - 6. inria-00110979

HAL Id: inria-00110979
https://inria.hal.science/inria-00110979

Submitted on 2 Nov 2006

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.

Reconnaissance de structures logiques par un réseau de neurones transparent

Rangoni Yves – Belaïd Abdel

Université de Nancy 2

LORIA - Equipe READ

Vandœuvre-lès-Nancy, France

{rangoni,abelaid}@loria.fr

Résumé :

Dans cet article, nous présentons une nouvelle approche à base de réseaux de neurones pour la reconnaissance de structures logiques de documents. Le système utilisé, appelé Réseau de Neurones Transparent, allie à la fois une approche dirigée par les données grâce à son apprentissage, et une approche dirigée par le modèle à l'aide d'une intégration dans sa topologie d'un modèle de classe. La reconnaissance se fait à travers des cycles d'interprétation-extraction avec une correction des entrées, permettant une classification des formes les plus ambiguës. L'application de ce système est effectuée sur une base de 74 articles scientifiques et un gain de 10 points de reconnaissance est obtenu en comparaison avec une approche classique (91.7% sur 21 structures logiques contre 81.6%). Après un rapide survol des différentes méthodes que nous propose la littérature, nous nous attacherons à la description de l'architecture du système. Puis, nous évoquerons sa capacité d'autocorrection lors de cycles perceptifs. Nous proposerons ensuite une méthode de partitionnement des données afin de réduire considérablement la tâche d'extraction de primitives physiques, et pour finir, nous présenterons les résultats et les perspectives obtenus par un tel système.

Mots-clés : Analyse d'Images de Documents, Réseau de Neurones Transparent, Perceptron Multicouches, Sélection de variables.

1 Introduction et problématique

Dans l'analyse d'images de documents (DIA), les applications des réseaux de neurones de type Perceptron Multicouches (MLP) se font essentiellement au niveau de la reconnaissance physique (*Physical Layout Analysis*), plutôt qu'au niveau de la structure logique (*Logical Layout Analysis*). On les retrouve donc plus généralement dans des tâches de traitement d'image en particulier au niveau du filtrage, de la suppression du bruit ainsi que pour la reconnaissance de caractères. Bien qu'il existe un grand nombre de travaux les mettant en scène, ils ont souvent été critiqués car, malgré des résultats prometteurs et intéressants, ils ont besoin d'un grand nombre d'exemples pour apprendre les relations entre physique et logique. La plupart des applications de DIA sont fondées sur des schémas traditionnels basés sur des perceptrons multicouches et ne proposent pas de nouvelles méthodologies [MAR 05]. Les contributions se font plutôt au niveau de la façon dont le MLP est utilisé pour résoudre une tâche spécifique.

A contrario, l'état de l'art nous montre que les systèmes à base de règles et de grammaires sont largement plus usités pour résoudre ce problème. Partant de la nature structurelle de la tâche à effectuer, ils sont a priori bien adaptés et conviennent assez bien à la plupart des problèmes. Cependant, ils ne se révèlent pas nécessairement efficaces pour traiter des documents complexes et bruités d'autant plus que les applications sur lesquelles sont testés ces systèmes sont souvent peu complexes, partent d'entrées très propres de haut niveau (*Symbol-based techniques*), et souvent les sorties à différentier sont peu nombreuses (4 ou 5 éléments dans la majorité des publications). Bien que des évolutions aient été apportées afin de gagner en souplesse et en généricité (logique floue, n-grammaires, etc. [MAO 03]) ces approches restent toutefois trop dépendantes du modèle et dans la majorité des cas, un grand nombre de règles et paramètres doivent être fixés par un expert, les rendant très peu flexibles aux variations, aux données bruitées et erronées.

Le système que nous proposons utilise à la fois une phase approche par les données en utilisant un apprentissage qui déterminera les relations entre éléments physiques et structures logiques et à la fois une approche dirigée par le modèle qui manquait aux MLP sera intégrée directement dans la topologie car comme évoqué dans [NAG 00] : une part de savoir (*Knowledge*) semble toutefois indispensable pour traiter ce genre de problème. De plus le processus de reconnaissance utilisera une approche mixte entre résolution «global vers local» et «local vers global» en simulant des cycles perceptifs empruntés à la perception humaine.

2 Architecture du système

L'architecture du système de reconnaissance se fonde sur un Perceptron Multi-Couches (MLP). À partir de ce MLP, nous donnons pour chaque neurone, y compris ceux des couches intermédiaires (ou couches cachées), une sémantique. Pour chaque neurone du réseau, nous connaissons la sortie attendue et cette sortie correspond à un concept interprétable. Ce réseau artificiel est appelé Réseau de Neurones Transparent (TNN). Il a déjà été utilisé dans le cadre de la reconnaissance de mots manuscrits (voir [CôT 95] pour un exemple de sémantique en 3 couches : indices physiques, lettres et mot). Dans l'analyse de documents, nous allons garder le même principe de décomposition de l'interprétation mais en attribuant aux couches successives des éléments de structures logiques de plus en plus généraux. La Fig.1 montre un exemple de décomposition qui peut être effectuée sur des documents de type article scientifique.

Le choix de la sémantique est dans notre cas assez aisé car il suffit de déployer une sorte de DTD (*Document Type Definition*) du document pour former des paliers d'interprétation. Même si une DTD n'est pas présente, la décomposition doit se faire du précis vers le général, la couche la plus à droite étant ce qu'il y a de plus «simple» à reconnaître. Nous nous sommes aussi aidé de la qualité générale des résultats des outils d'extraction pour placer les neurones. Notons qu'à part la couche des sorties attendues qui se veut spécifique à la classe de document, les couches suivantes sont largement générales pour couvrir n'importe quel document de type éditorial.

Dans ce réseau, la sortie désirée est placée dès la deuxième couche, les suivantes servant à apporter du contexte. L'apprentissage du réseau se fait localement entre chaque paire de couches successives. On peut

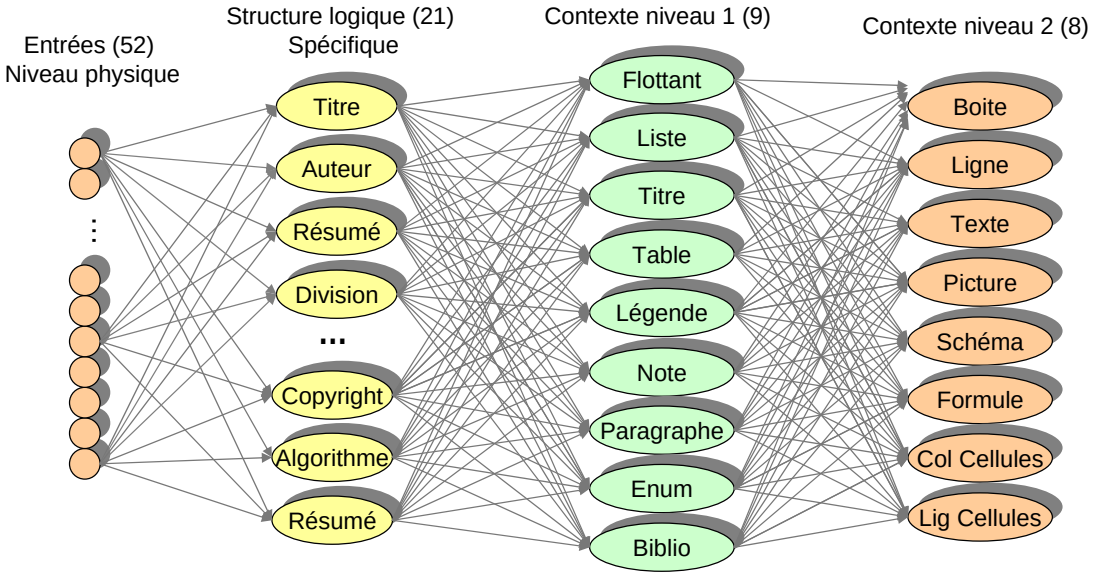


FIG. 1: Topologie de réseau de neurones transparent pour la reconnaissance d'articles scientifiques

voir le système comme une cascade de réseaux sans couches cachées. Utilisé tel quel, ce réseau est moins bon qu'un MLP car il est bridé lors de son apprentissage (les activations attendues des couches intermédiaires sont données et non plus estimées), mais se révèle plus efficace couplé avec des cycles perceptifs.

3 Cycles perceptifs

3.1 Principe

Les cycles perceptifs ont pour but une correction du vecteur d'entrée composé des différents indices pris au niveau physique (Fig. 1). Dans un premier temps, l'information est propagée dans le réseau puis la sortie calculée O est analysée : si le vecteur O est proche d'un vecteur unité, on considère que la forme est classifiée sinon elle est rejetée.

$$\|O\|_\infty > \varepsilon \qquad \text{avec} \qquad 0 \ll \varepsilon < 1$$

$$\Gamma(O) = \frac{n((\sum O_i)^2 - \sum O_i^2)}{(n-1)(\sum O_i)^2} < \eta \qquad \text{avec} \qquad 0 < \eta \ll 1$$

Avec un bon choix de ε et η (η toutefois grand car les sorties des réseaux de neurones ne sont jamais parfaites), les conditions sont plus fortes que dans la plupart des applications utilisant des MLP qui ne considèrent que $argmax(O)$ ce qui biaise les résultats. Si la forme est rejetée, on remonte les liens en cherchant les composantes I_i du vecteur d'entrée qui ont pu poser problème. Elles sont triées dans une liste par ordre décroissant d'influence en attente de correction. Cette correction peut se faire directement en paramétrant à nouveau les algorithmes (voire les changer par des plus performants) qui ont donné ces composantes en suivant la liste crée. Plus généralement, on utilise les activations des neurones des couches de contexte (normalement avec moins d'erreur) pour apporter de l'information sur la forme. D'autres listes d'hypothèses sont ainsi crées en ordonnant par vraisemblance la réelle nature de la forme à reconnaître. Dans le cas de l'analyse de document, cette technique est particulièrement efficace car les erreurs liées à la segmentation

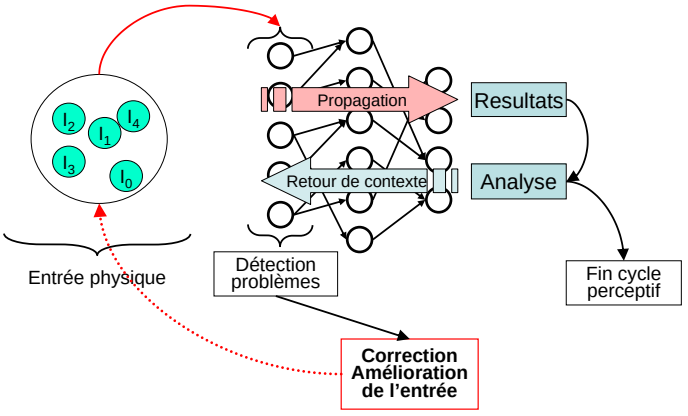


FIG. 2: Reconnaissance par cycles perceptifs

sont la cause de la majeure partie des problèmes directs ou indirects. Grâce à l'information de contexte, on peut émettre des hypothèses sur la forme et décider de revenir sur l'entrée en donnant par exemple une nouvelle boîte englobante adaptée à l'hypothèse. Si les hypothèses sont bien ordonnées, au bout d'un certain nombre de cycles, il est possible de récupérer une partie du rejet tout en gardant le même modèle et les mêmes outils. La Fig. 2 schématise le processus de reconnaissance.

Pour chaque forme F_i à reconnaître **faire**
  Calculer les activations de tous les neurones
  Si (F_i reconnue) **Alors**
    [Rien à faire et quitter]
  Fin Si
  $L_n \leftarrow$Ordonner neurones de la dernière couche C_n par activation décroissante
  Pour chaque neurone N_j de L_n **faire**
    $L_{n-1,j} \leftarrow$Ordonner neurones de la dernière couche C_{n-1} par activation décroissante et par importance de lien décroissant
    Pour chaque neurone N_k de $L_{n-1,j}$ **faire**
      ...
    Fin Pour
  Fin Pour
  En parcourant dans l'ordre l'arbre des hypothèses formé par les $L_{x,y}$
  Dérouler les hypothèses et corriger en conséquence
  Relancer la procédure avec forme corrigée
Fin Pour

Algorithme 1: Utilisation du contexte

3.2 Correction des entrées

La correction se fait en comparant la sortie obtenue et un échantillon type (ce que l'on aurait dû avoir). L'échantillon type est choisi suivant l'hypothèse émise sur la forme à reconnaître et chaque échantillon est calculé à partir de la base d'apprentissage. Il y a plusieurs façons de déterminer les échantillons types pour chaque classe mais malheureusement aucune méthode exacte. Plusieurs approches ont été testées pour trouver «mathématiquement» le meilleur échantillon mais le résultat obtenu, bien que juste pour le classifieur, ne donne pas un échantillon «réaliste» et interprétable, dans le pire des cas corriger à partir de celui-ci donnait des résultats plus mauvais qu'en ne corrigeant pas. Nous avons choisi dans un premier temps de prendre l'échantillon le plus proche de la moyenne des échantillons comme base de comparaison. C'est d'ailleurs cette méthode qui a été retenue pour les tests de la section 6. D'autres solutions sont encore à l'étude pour trouver de meilleurs représentants et si possible déterminés en prétraitement.

3.3 Problème des cycles perceptifs

Le réseau de neurones transparent montre ses avantages lorsqu'il est couplé à des cycles perceptifs, la transparence du réseau étant un avantage pour les cycles perceptifs lors de la correction. Bien que le système soit prévu pour limiter le nombre de cycles, il est toujours possible de boucler un certain temps sur des formes particulièrement difficiles à reconnaître. La propagation dans le réseau étant insignifiante, le fait d'extraire plusieurs fois des primitives de haut niveau (nécessitant donc un temps d'extraction élevé) peut se révéler lourd lors de la reconnaissance. Afin de limiter ce temps, il est possible de le raccourcir sur les formes plus simples et garder les outils les plus gourmands en réserve pour les formes à problèmes.

4 Partitionnement de l'espace d'entrée

4.1 Reconnaissance avec des entrées réduites

Comme dans tous les systèmes, la première extraction est incompressible car il faut bien posséder les entrées pour calculer la sortie, mais dans notre cas, de part la présence de cycles perceptifs, nous sommes amenés à effectuer cette extraction plusieurs fois (autant que de cycles perceptifs nécessaires). Au pire des cas, la reconnaissance est d'autant de fois plus longue qu'il faut de cycles perceptifs pour reconnaître les formes. Il y a toutefois des formes simples à reconnaître, et qui sont classifiées parfois même au bout d'un seul cycle. Afin d'accélérer encore plus ce processus, nous avons fait le choix de partitionner l'espace d'entrée en quelques groupes qui sont donnés progressivement au réseau (Fig. 3). À partir de l'ensemble du vecteur d'entrée, plusieurs groupes de variables disjoints sont formés. Le but de ce découpage est de donner, si possible, un nombre plus faible de variables en entrée du TNN, afin de réduire le temps global de reconnaissance. Le réseau est donc alimenté par un petit groupe de variables dans un premier temps ; si ce groupe suffit à reconnaître la forme, le processus s'arrête, dans le cas contraire on effectue une correction de ce groupe comme décrit dans le 3.2. Si au bout de plusieurs cycles perceptifs le résultat n'est toujours pas satisfaisant, on

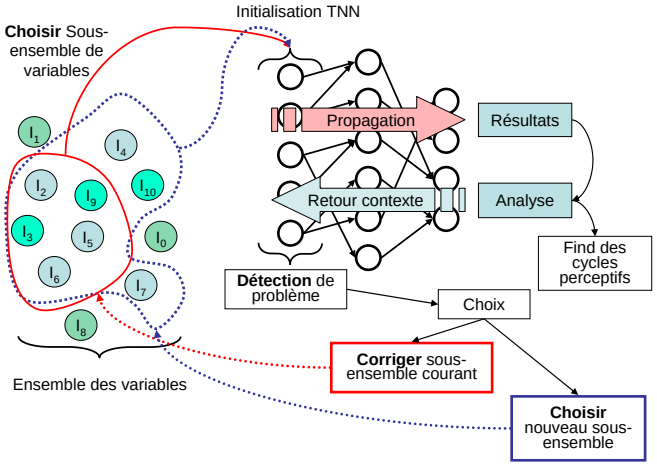


FIG. 3: Cycles perceptifs avec groupes de variables

change la taille du vecteur en lui rajoutant les variables du deuxième groupe et on refait une série de cycles perceptifs et ainsi de suite jusqu'à utiliser éventuellement toutes les variables. L'idée est de ne donner que ce dont à besoin le réseau pour reconnaître les formes en adaptant la quantité de travail à fournir en fonction de la difficulté de la forme à classifier.

4.2 Création de groupes de variables

Pour créer ces groupes, les variables peuvent être évaluées suivant deux critères : la rapidité d'extraction et la qualité informative. Dans le premier cas, il est assez facile de quantifier ce critère car il suffit de prendre un temps d'exécution ou alors utiliser la complexité de l'algorithme qui permet d'obtenir la variable. Dans le second cas, évaluer le pouvoir informatif d'une variable n'a rien de trivial. En effet, aucune méthode (non combinatoire) n'arrive à prédire exactement l'importance de chaque variable pour le classifieur. Pour créer de bons groupes suivant l'importance, nous nous sommes intéressés à la sélection de variables.

Il existe dans la littérature deux grande familles d'approche : les approches par filtres et la approches «embarquée» (*Wrapper*) [GUY 03]. Dans notre cas les deux familles de méthode sont intéressantes mais ne conviennent pas seules au problème posé. En effet les méthodes à base de filtre considèrent les variables indépendamment les unes des autres et risquent de donner des groupes dont la redondance à l'intérieur de chacun peut être forte et par conséquent donner de faibles résultats une fois passés dans un réseau de neurones. Les méthodes embarquées utilisent le classifieur pour sélectionner les variables, elles sont bien plus puissantes mais nécessitent que le classifieur ait déjà été construit pour pouvoir les mettre en œuvre et dans notre cas, nous avons besoin des groupes de variables pour pouvoir construire les différents classifieurs.

La méthode que nous proposons part d'une méthode à base de filtre tout en traitant les variables les une en fonction des autres afin de limiter la redondance au sein d'un même groupe. L'algorithme suivant résume les différentes étapes de la création du partitionnement.

Comme dans la grande majorité des approches, nous commençons le calcul de la matrice de corrélation de la base de données qui est décomposée en valeurs propres et vecteurs propres. La base formée par les vecteurs propres donne les axes maximisant l'inertie du nuage (variables) projeté sur cet axe. En choisissant les q premières composantes des vecteurs propres correspondant au q premières valeurs propres rangées en ordre décroissant, ce nouveau espace de plus petite dimension permet de réécrire les variables de départ en perdant le moins d'erreur possible (au sens des moindres carrés).

Dans notre cas, nous avons besoin de garder les variables d'origine pour pouvoir effectuer les corrections, donc au lieu de créer de nouvelles variables par combinaison linéaire des autres comme dans l'ACP (Ana-

lyse en Composantes principales), nous partons du fait que si deux variables sont corrélées (ou partagent de l'information commune) alors les vecteurs propres correspondants sont très proches (en valeur absolue). De cette constatation, on peut alors regrouper les vecteurs en utilisant une méthode de clustering comme un k-means ou une carte auto-organisatrice [KOH 88].

$X \leftarrow$base des échantillons
$m \leftarrow Card(X)$
$n \leftarrow dim(X[i])$
$Cor \leftarrow (r_{i,j})_{i,j\in[1,n]}$ avec $r_{i,j} = corr(X_i, X_j) = \frac{\frac{1}{m}\sum_{k=1}^{m}(X[i]_k - \bar{X[i]})(X[j]_i - \bar{X[j]})}{\sqrt{\frac{1}{m}\sum_{k=1}^{m}(X[i]_k - \bar{X[i]})^2}\sqrt{\frac{1}{m}\sum_{k=1}^{m}(X[j]_k - \bar{X[j]})^2}}$
$V \leftarrow vecteurs_propres(Cor)$
$\Lambda \leftarrow diag(sort(valeurs_propres(Cor), >))$
$V \times \Lambda \times V^T \leftarrow Cor$
$V^q \leftarrow q$ première colonnes de V avec $q \leftarrow Cattell(\Lambda)$
Clusteriser les $|V_i|, i \in [1, n]$ en p clusters C_j
$j \leftarrow 1$
Tant que (Il reste des variables x_i) **faire**
    Pour Chaque C_j **faire**
        Chercher le V_i le plus proche du centre de C_j
        $P_j \leftarrow$L'ensemble des x_i correspondant aux V_i trouvés
    Fin Pour
    $j \leftarrow j + 1$
Fait

Algorithme 2: Création d'un partitionnement des variables d'entrée

Les clusters regroupent des vecteurs de «même nature» et à l'intérieur de chaque, le vecteur le plus proche du centre est celui qui explique au mieux le groupe. Si l'on crée alors un ensemble constitué des meilleurs représentants de chaque groupe, on a à la fois des variables informatives et qui donnent de l'information différente (variables indépendantes). Les problèmes de redondance sont ainsi minimisés, il suffit de réitérer le processus pour obtenir plusieurs groupes de variables jusqu'à partitionner l'ensemble de l'espace d'entrée. Cette méthode automatique a été évaluée suivant ce protocole : on utilise un MLP comme classifieur qui lui-même utilise les variables à traiter et des sorties attendues. La topologie est choisie pour avoir si possible le meilleur taux de reconnaissance normalisé en utilisant toutes les variables. Nous créons ensuite à l'aide la précédente méthode les variables du premier groupe qui seront présentées seules à un MLP de même topologie que l'ancien et nous comparons les taux de reconnaissance pour évaluer la perte d'information. La comparaison de la méthode se fait avec le meilleur résultat parmi 1000 groupes crées aléatoirement (Fig.4). Trois tests sont effectués : les deux premiers sur une base d'images [LEC] pour traiter des variables de bas-niveau et le troisième sur la base des variables physiques (haut-niveau) extrait de documents scientifiques.

#Variables	Méthode	
	Aléatoire	Notre sélection
784 (max)	100%	100%
500	98.4%	99.2%
300	95.9%	98.4%
150	90.5%	96.5%
100	84.2%	94.2%
50	70.9%	87.8%
25	47.1%	67.6%

Base MNIST, images 28×28 pixels

#Variables	Méthode	
	Aléatoire	Notre sélection
49 (max)	100%	100%
35	94.2%	99.3%
25	81.2%	88.6%
15	56.2%	70.5%
10	43.9%	55.2%

Base MNIST, images rééchantillonnées en 7×7 pixels

#Variables	Méthode	
	Aléatoire	Notre sélection
56 (max)	100%	100%
35	86.9%	99.3%
25	65.0%	79.6%
15	51.8%	80.1%
10	35.1%	83.8%
5	17.9%	44.9%

Base d'indices physiques Siggraph

FIG. 4: Information gardée par la méthode de création de groupe en fonction du nombre de variables

Pour le choix de la dimension du sous-espace à considérer, plusieurs méthodes ont été investiguées. Nous avons retenu l'approche de Cattell [CAT 66] qui consiste à trouver le «coude» dans la courbe formée en prenant les numéros des valeurs propres en abscisse et leurs valeurs respectives en ordonnée. Dans cette méthode de partitionnement, il n'y a qu'un paramètre à faire varier (le nombre de groupes souhaité) et des initialisations aléatoires.

Durant les tests, le choix de la dimension du sous-espace influe finalement relativement peu dans la création des groupes (la méthode de Cattell est utilisée pour tous les tests). Même si le premier meilleur groupe n'est que sous-optimal, les variables manquantes sont tout de même dans le groupe suivant et seront donc au pire utilisé lors des cycles perceptifs.

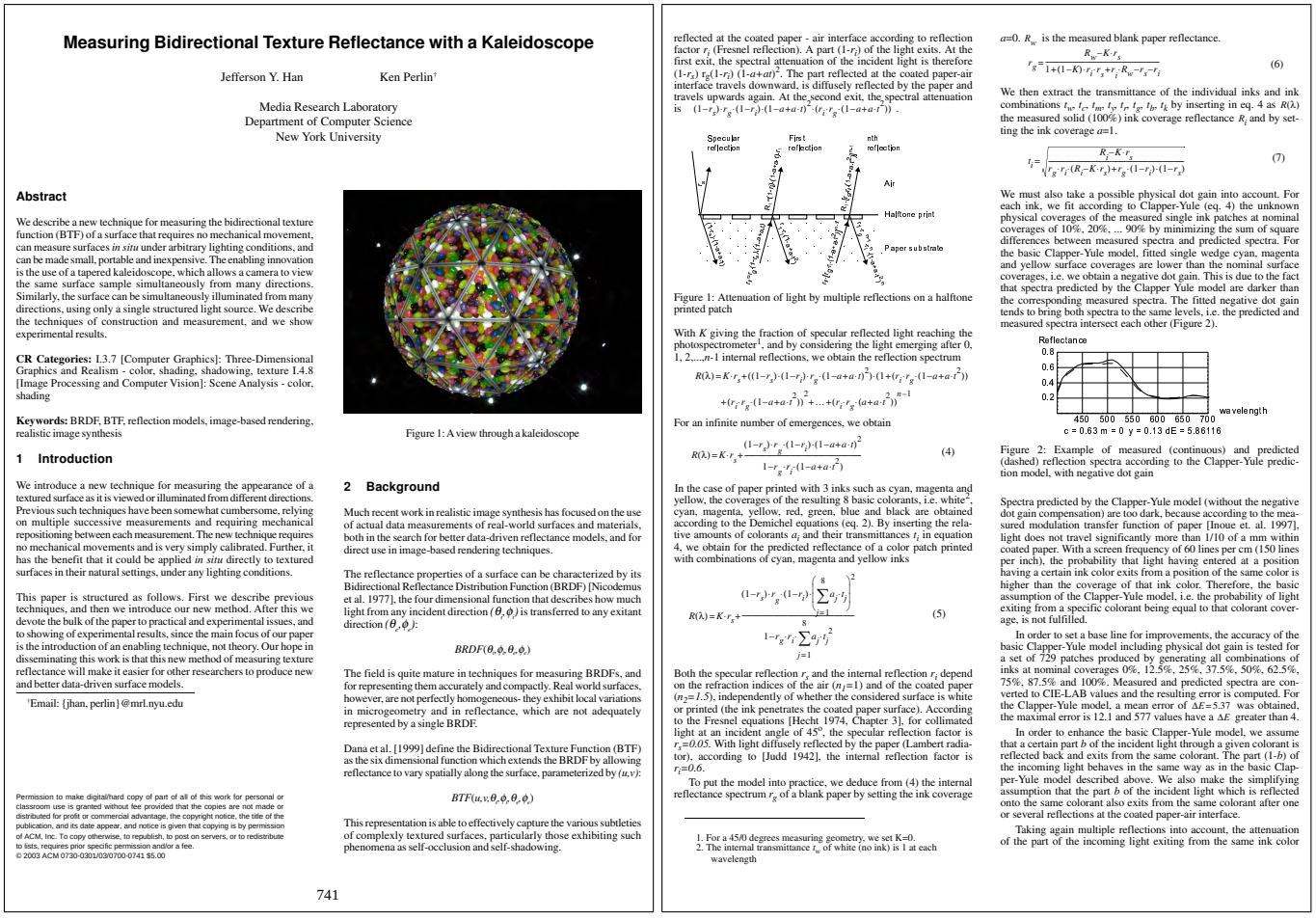
5 Résultats et discussion

Pour effectuer les tests suivants, une base d'article scientifique a été utilisée. Il s'agit d'articles provenant de la conférence Siggraph [SIG 03] qui dispose d'une mise en page et d'éléments logiques assez fournis (Fig. 5). À partir de 74 documents de cette base, nous avons étiquetés 21 structures logiques, ce qui représente un peu plus de 2000 formes à reconnaître au total. La base est coupée en deux parties, d'un coté 44 documents pour l'apprentissage et de l'autre 30 pour le test.

Les entrées (indices physiques) sont données en grande partie par un moteur d'OCR (plutôt les variables de type géométrique et morphologiques), le reste provenant de nos propres procédures (plutôt les variables «sémantiques»). Le tableau 1 montre l'ensemble des entrées/sorties utilisées. Après avoir déterminé par la méthode de partitionnement automatique trois groupes de variables, trois réseaux de neurones transparents

Éléments logiques		Title, Author, Email, Locality, Abstract, Key words, CR Categories, Introduction, Paragraph, Section, SubSection, SubSubSection, List, Enumeration, Float, Conclusion, Bibliography, Algorithms, Copyright, Acknowledgments, Page number
Physique	Géométrie	Text, Image, Table, Other, x position, y position, Width, Height, NumPage, UpSpace, BottomSpace, LeftSpace, RightSpace
	Morphologie	Bold, Italic, Underlined, Strikethrough, UpperCase, Small Capitals, Subscript, Superscript, Font Name, Font Size, Scaling, Spacing, Alignment, LeftIndent, RightIndent, FirstIndent, NumLines, Boxed, Red/Green/Blue
	Sémantique	IsNumeric, KeyWords, %KnownWords, %Punctuation, Bullet, Enum, Language, Baseline

TAB. 1: Ensemble des variables physiques et de éléments logiques utilisés pour les tests



Measuring Bidirectional Texture Reflectance with a Kaleidoscope

Jefferson Y. Han Ken Perlin†

Media Research Laboratory
Department of Computer Science
New York University

Abstract

We describe a new technique for measuring the bidirectional texture function (BTF) of a surface that requires no mechanical movement, can measure surfaces *in situ* under arbitrary lighting conditions, and can be made small, portable and inexpensive. The enabling innovation is the use of a tapered kaleidoscope, which allows a camera to view the same surface sample simultaneously from many directions. Similarly, the surface can be simultaneously illuminated from many directions, using only a single structured light source. We describe the techniques of construction and measurement, and we show experimental results.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - color, shading, shadowing, texture I.4.8 [Image Processing and Computer Vision]: Scene Analysis - color, shading

Keywords: BRDF, BTF, reflection models, image-based rendering, realistic image synthesis

1 Introduction

We introduce a new technique for measuring the appearance of a textured surface as it is viewed or illuminated from different directions. Previous such techniques have been somewhat cumbersome, relying on multiple successive measurements and requiring mechanical repositioning between each measurement. The new technique requires no mechanical movements and is very simply calibrated. Further, it has the benefit that it could be applied *in situ* directly to textured surfaces in their natural settings, under any lighting conditions.

This paper is structured as follows. First we describe previous techniques, and then we introduce our new method. After this we devote the bulk of the paper to practical and experimental issues, and to showing of experimental results, since the main focus of our paper is the introduction of an enabling technique, not theory. Our hope in disseminating this work is that this new method of measuring texture reflectance will make it easier for other researchers to produce new and better data-driven surface models.

†Email: {jhan, perlin}@mrl.nyu.edu

Permission to make digital/hard copy of part of all of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.
© 2003 ACM 0730-0301/03/0700-0741 $5.00

Figure 1: A view through a kaleidoscope

2 Background

Much recent work in realistic image synthesis has focused on the use of actual data measurements of real-world surfaces and materials, both in the search for better data-driven reflectance models, and for direct use in image-based rendering techniques.

The reflectance properties of a surface can be characterized by its Bidirectional Reflectance Distribution Function (BRDF) [Nicodemus et al. 1977], the four dimensional function that describes how much light from any incident direction (θ_i, ϕ_i) is transferred to any exitant direction (θ_e, ϕ_e):

$$BRDF(\theta_i, \phi_i, \theta_e, \phi_e)$$

The field is quite mature in techniques for measuring BRDFs, and for representing them accurately and compactly. Real world surfaces, however, are not perfectly homogeneous- they exhibit local variations in microgeometry and in reflectance, which are not adequately represented by a single BRDF.

Dana et al. [1999] define the Bidirectional Texture Function (BTF) as the six dimensional function which extends the BRDF by allowing reflectance to vary spatially along the surface, parameterized by (u,v):

$$BTF(u, v, \theta_i, \phi_i, \theta_e, \phi_e)$$

This representation is able to effectively capture the various subtleties of complexly textured surfaces, particularly those exhibiting such phenomena as self-occlusion and self-shadowing.

reflected at the coated paper - air interface according to reflection factor r_i (Fresnel reflection). A part $(1\text{-}r_i)$ of the light exits. At the first exit, the spectral attenuation of the incident light is therefore $(1\text{-}r_s)\ t_g(1\text{-}r_i)\ (1\text{-}a+at)^2$. The part reflected at the coated paper-air interface travels downward, is diffusely reflected by the paper and travels upwards again. At the second exit, the spectral attenuation is $(1-r_s)\cdot r_g\cdot(1-r_i)\cdot(1-a+a\cdot t)^2\cdot(r_i\cdot r_g\cdot(1-a+a\cdot t^2))$.

Figure 1: Attenuation of light by multiple reflections on a halftone printed patch

With K giving the fraction of specular reflected light reaching the photospectrometer[1], and by considering the light emerging after 0, 1, 2,...,n-1 internal reflections, we obtain the reflection spectrum

$$R(\lambda)=K\cdot r_s+((1-r_s)\cdot(1-r_i)\cdot r_g\cdot(1-a+a\cdot t)^2)\cdot(1+(r_i\cdot r_g\cdot(1-a+a\cdot t^2))+(r_i\cdot r_g\cdot(1-a+a\cdot t^2))^2+\ldots+(r_i\cdot r_g\cdot(a+a\cdot t^2))^{n-1})$$

For an infinite number of emergences, we obtain

$$R(\lambda)=K\cdot r_s+\frac{(1-r_s)\cdot r_g\cdot(1-r_i)\cdot(1-a+a\cdot t)^2}{1-r_g\cdot r_i\cdot(1-a+a\cdot t^2)} \quad (4)$$

In the case of paper printed with 3 inks such as cyan, magenta and yellow, the coverages of the resulting 8 basic colorants, i.e. white[2], cyan, magenta, yellow, red, green, blue and black are obtained according to the Demichel equations (eq. 2). By inserting the relative amounts of colorants a_j and their transmittances t_j in equation 4, we obtain for the predicted reflectance of a color patch printed with combinations of cyan, magenta and yellow inks

$$R(\lambda)=K\cdot r_s+\frac{(1-r_s)\cdot r_g\cdot(1-r_i)\left(\sum_{j=1}^{8}a_j\cdot t_j\right)^2}{1-r_g\cdot r_i\cdot\sum_{j=1}^{8}a_j\cdot t_j^2} \quad (5)$$

Both the specular reflection r_s and the internal reflection r_i depend on the refraction indices of the air (n_2=1) and of the coated paper (n_2=*1.5*), independently of whether the considered surface is white or printed (the ink penetrates the coated paper surface). According to the Fresnel equations [Hecht 1974, Chapter 3], for collimated light at an incident angle of 45°, the specular reflection factor is r_s=*0.05*. With light diffusely reflected by the paper (Lambert radiator), according to [Judd 1942], the internal reflection factor is r_i=*0.6*.

To put the model into practice, we deduce from (4) the internal reflectance spectrum r_g of a blank paper by setting the ink coverage a=0. R_w is the measured blank paper reflectance.

$$r_g=\frac{R_w-K\cdot r_s}{1+(1-K)\cdot r_i\cdot r_s+r_i\cdot R_w-r_s-r_i} \quad (6)$$

We then extract the transmittance of the individual inks and ink combinations t_w, t_c, t_m, t_y, t_r, t_g, t_b, t_k by inserting in eq. 4 as $R(\lambda)$ the measured solid (100%) ink coverage reflectance R_i and by setting the ink coverage a=1.

$$t_i=\sqrt{\frac{R_i-K\cdot r_s}{r_g\cdot r_i\cdot(R_i-K\cdot r_s)+r_g\cdot(1-r_i)\cdot(1-r_s)}} \quad (7)$$

We must also take a possible physical dot gain into account. For each ink, we fit according to Clapper-Yule (eq. 4) the unknown physical coverages of the measured single ink patches at nominal coverages of 10%, 20%, ... 90% by minimizing the sum of square differences between measured spectra and predicted spectra. For the basic Clapper-Yule model, fitted single wedge cyan, magenta and yellow surface coverages are lower than the nominal surface coverages, i.e. we obtain a negative dot gain. This is due to the fact that spectra predicted by the Clapper Yule model are darker than the corresponding measured spectra. The fitted negative dot gain tends to bring both spectra to the same levels, i.e. the predicted and measured spectra intersect each other (Figure 2).

Figure 2: Example of measured (continuous) and predicted (dashed) reflection spectra according to the Clapper-Yule prediction model, with negative dot gain

Spectra predicted by the Clapper-Yule model (without the negative dot gain compensation) are too dark, because according to the measured modulation transfer function of paper [Inoue et. al. 1997], light does not travel significantly more than 1/10 of a mm within coated paper. With a screen frequency of 60 lines per cm (150 lines per inch), the probability that light having entered at a position having a certain ink color exits from a position of the same color is higher than the coverage of that ink color. Therefore, the basic assumption of the Clapper-Yule model, i.e. the probability of light exiting from a specific colorant being equal to that colorant coverage, is not fulfilled.

In order to set a base line for improvements, the accuracy of the basic Clapper-Yule model including physical dot gain is tested for a set of 729 patches produced by generating all combinations of inks at nominal coverages 0%, 12.5%, 25%, 37.5%, 50%, 62.5%, 75%, 87.5% and 100%. Measured and predicted spectra are converted to CIE-LAB values and the resulting error is computed. For the Clapper-Yule model, a mean error of ΔE=5.37 was obtained, the maximal error is 12.1 and 577 values have a ΔE greater than 4.

In order to enhance the basic Clapper-Yule model, we assume that a certain part b of the incident light through a given colorant is reflected back and exits from the same colorant. The part (1-b) of the incoming light behaves in the same way as in the basic Clapper-Yule model described above. We also make the simplifying assumption that the part b of the incident light which is reflected onto the same colorant also exits from the same colorant after one or several reflections at the coated paper-air interface.

Taking again multiple reflections into account, the attenuation of the part of the incoming light exiting from the same ink color

1. For a 45/0 degrees measuring geometry, we set K=0.
2. The internal transmittance t_w of white (no ink) is 1 at each wavelength

FIG. 5: Exemple de deux pages extraitres de la base de données

sont ainsi appris, chaque réseau correspondant à la taille d'entrée différente de chaque groupe. La reconnaissance s'effectue comme décrit dans la section 2 en fixant par avance le nombre maximum de cycles perceptifs à 4. Le tableau 2 montre les résultats de reconnaissance du systèmes à chaque cycle perceptifs le dernier étant gardé comme résultat final. Les taux de reconnaissance sont comparés à un perceptron multicouche classique possédant 2 couches cachées de 50 et 30 neurones respectivement.

Au bout du 1er cycles perceptif, le TNN est bien évidemment moins performant que le MLP car ce dernier

Classes prises en compte	MLP	TNN C_1	C_2	C_3	C_4
Toutes les classes	81.6%	45.2	78.9	90.2	91.7%
Classe la mieux reconnue	86.9%	66.7	85.3	85.3	99.3%
Classe la moins bien reconnue	0.0%	0.0	0.0	4.0	28.6%
Facteur de temps (le MLP étant la référence)	1	0.7	1.45	1.85	2.40

TAB. 2: Classification de structures logiques par un MLP et par un TNN avec cycles perceptifs

possédant des couches cachées à moins de contraintes pour reconnaître les formes. Notons aussi qu'au 1er cycle perceptif, le TNN ne possède qu'un tiers des variables (le MLP les a toutes) et donne cependant des résultats honorables avec un temps moindre. Au deuxième et troisième cycle, le taux de reconaissance passe de 45.2% à 90.2% en allongeant d'un peu moins de deux fois le temps de calcul. À la fin du quatrième cycle perceptif, le TNN prend largement le dessus grâce à son principe d'interprétation-correction en gagnant plus de 10 points de reconnaissance (de 81.6% à 91.7%). Bien sûr, cette augmentation du taux s'accompagne inévitablement d'un allongement du temps de calcul qui se voit multiplié par 2.4. En éliminant tôt les formes faciles à reconnaître et en utilisant progressivement des groupes de variables, on arrive ainsi à gagner en précision sans pour autant multiplier par 4 le temps de reconnaissance. Les principales erreurs corrigées proviennent de la segmentation et le fait de pouvoir les corriger permet d'expliquer presque entièrement l'amélioration de la reconnaissance. Les formes rejetées par le TNN sont pratiquement toutes rejetées par le MLP et correspondent très souvent à des éléments de structures peu fréquents et/ou ne possédant pas une apparence physique très distinguable. On peut même noter que pour la classe la moins reconnue, le TNN arrive à «sauver» 28.6% des éléments alors que le MLP n'en avait reconnu aucun. Un cinquième cycle, dans ce cas applicatif, n'apporte presque rien au niveau de la reconnaissance et allonge par contre considérablement le temps de calcul. Notons que la phase d'apprentissage est extrêmement rapide (quelques minutes sur un P4 3 Ghz) et n'a ni besoin d'un grand nombre d'échantillons ni d'un grand nombre d'époques pour se stabiliser.

6 Conclusions et perspectives

Dans cet article nous avons présenté une approche hybride qui permet à la fois de bénéficier des avantages d'une méthode basée sur les données en utilisant un réseau de neurones artificiel et une méthode basé sur l'intégration de savoir en insérant des concepts dans les couches cachées du réseau. Le système nommé

réseau de neurones transparent utilise à la fois une approche descendante et une approche montante lors de la reconnaissance modélisée par des cycles perceptifs qui consiste à réitérer des suites de vision - interprétation - correction facilité par un retour de contexte. Le système proposé a été testé dans le cadre de la reconnaissance de structures logiques de documents. Les résultats montrent qu'un tel système, bien que basé sur un réseau de neurones, permet d'obtenir un bon taux de reconnaissance et permet de mieux prendre en compte et d'expliquer à la fois les échantillons reconnus comme ceux rejetés.

Dans les prochains travaux, nous essayons d'autres topologies que celle présenté dans cet article. Nous allons introduire une ou plusieurs couches cachées entre les entrées et les sorties pour gagner encore un peu plus de souplesse quitte à perdre de l'information lors des retour de contexte. L'utilisation ou plutôt l'extension de ce réseau à une version de type récurrente sera elle aussi étudié et devrait même permettre de renforcer l'information lors des retours.

Références

[CAT 66] CATTELL R. B., The scree test for the number of factors, *Multivariate Behavioral Research*, 1966, pp. 245-276.

[CôT 95] CÔTÉ M., Building a Perception Based Model for Reading Cursive Script, *International Conference on Document Analysis and Recognition*, 1995, pp. 75-76.

[GUY 03] GUYON I., ELISSEEFF A., An introduction to variable and feature selection, *Machine Learning Research*, 2003, pp. 1157-1182.

[KOH 88] KOHONEN T., Self-organization and associative memory, *Multivariate Behavioral Research, Springer-Verlag*, 1988.

[LEC] LECUN Y., http ://yann.lecun.com/exdb/mnist/.

[MAO 03] MAO S., ROSENFELD A., KANUNGO T., Document Structure Analysis Algorithms : A Literature Survey, *SPIE Electronic Imaging*, 2003, pp. 197-207.

[MAR 05] MARINAI S., GORI M., SODA G., Artificial Neural Networks for Document Analysis and Recognition, *Pattern Analysis and Machine Intelligence*, 2005, pp. 23-35.

[NAG 00] NAGY G., Twenty years of document image analysis in PAMI, *Pattern Analysis and Machine Intelligence*, 2000, pp. 38-62.

[SIG 03] SIGGRAPH, http ://www.siggraph.org/s2003/, 2003.